5/11.00

# WEST

Help | Logout

Main Menu | Search Form | Posting Counts | Show S Numbers | Edit S Numbers

**Search Results -**

| Terms | Documents |
|---|---|
| l1 and domain and domain and domain | 20 |

**Database:** US Patents Full-Text Database

Refine Search: l1

**Search History**

| DB Name | Query | Hit Count | Set Name |
|---|---|---|---|
| USPT | l1 and domain and domain and domain | 20 | L4 |
| USPT | l1 and domain$1 | 20 | L3 |
| USPT | 1 and domain$1 | 52776 | L2 |
| USPT | #S653 | 138 | L1 |

5/11.00



## Document Number 8

Entry 8 of 20 File: USPT Jan 5, 1999

US-PAT-NO: 5857184
DOCUMENT-IDENTIFIER: US 5857184 A

TITLE: Language and method for creating, organizing, and retrieving data from a database
DATE-ISSUED: January 5, 1999

INVENTOR-INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY |
|---|---|---|---|---|
| Lynch; Jo | Cambridge | MA | N/A | N/A |

ASSIGNEE INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY | TYPE CODE |
|---|---|---|---|---|---|
| Walden Media, Inc. | Boston | MA | N/A | N/A | 02 |

APPL-NO: 8/ 646724
DATE FILED: May 3, 1996

INT-CL: [6] G06F 17/30
US-CL-ISSUED: 707/4; 707/1, 707/3, 707/102
US-CL-CURRENT: 707/4; 707/1, 707/102, 707/3
FIELD-OF-SEARCH: 707/1, 707/2, 707/3, 707/4, 707/5, 707/102

REF-CITED:

U.S. PATENT DOCUMENTS

| PAT-NO | ISSUE-DATE | PATENTEE-NAME | US-CL |
|---|---|---|---|
| 5197005 | March 1993 | Shwartz et al. | 707/2 |
| 5454101 | September 1995 | Mackay et al. | 707/3 |
| 5550971 | August 1996 | Brunner et al. | 707/3 |
| 5557794 | September 1996 | Matsunaga et al. | 707/3 |
| 5659727 | August 1997 | Velissaropoulos et al. | 707/2 |
| 5664173 | September 1997 | Fast | 707/4 |
| 5749079 | May 1998 | Yong et al. | 707/100 |

ART-UNIT: 276
PRIMARY-EXAMINER: Black; Thomas G.
ASSISTANT-EXAMINER: Robinson; Greta L.
ATTY-AGENT-FIRM: White; Mark P.

ABSTRACT:

A database system comprises a novel process for the organization, storage and retrieval of amorphous or ordered data. The data is

organized into data threads, each thread comprising an address code which characterizes and describes the data, and the data body. The address code is organized in a hierarchy having a syntax and morphology similar to the structure of human languages. The language, called REMDL, contains nouns, verbs, relational modifiers, operators, punctuation, and literal strings. The nouns, which signify events, are modified, by means of relational modifier, to form noun declensions, while the verbs, which signify actions, when modified, form verb conjugations. REMDL further contains rules for the time-relationship between events by means of derivatives. By use of this language, data may be quickly and efficiently stored and may be retrieved with a high level of flexibility.

15 Claims, 6 Drawing figures

Document Number 8

Entry 8 of 20 File: USPT Jan 5, 1999

DOCUMENT-IDENTIFIER: US 5857184 A
TITLE: Language and method for creating, organizing, and retrieving data from a database

BSPR:
We live in the information age. Information is essential for the management of most business ventures, especially in domains such as media and marketing. Information is nearly worthless, however, unless it is organized in a way which allows for simple and rapid retrieval of the desired portions of this information. Data organized in a computer in ways amenable to rapid and comprehensive retrieval is generally referred to as a "database".

BSPR:
As an example, consider a model which is used to manage a parts ordering system. The "entities" in such a system would include both parts and orders. One table may be constructed to show parts, and a second to show orders. A third table may then be generated to show the "relationship" between the parts and the orders.

BSPR:
Such a database model is appropriate for situations in which the data is easily categorized into a small, finite number of categories: parts, orders, prices, dates, etc.

BSPR:
The address code phrase characterizes the data to follow in a hierarchical manner, starting with the most general characteristic of the data, and descending to the most particular characteristic. The hierarchy chosen mimics a human language, organizing the code into nouns, verbs, and other grammatical types., wherein the nouns and verbs may have modified forms in order to characterize the data with greater specificity. This "meta language" is called REMDL.

BSPR:
Activities, as represented by verbs, include such items as USAGE, which is represented in REMDL by the term "DC". Usage is a broad domain verb phraseology which may comprise such items as:

BSPR:
According to one aspect of the invention, a database is organized into data threads wherein each data thread comprises an address code and a data body. The address code characterizes the data in the data body and comprises a multiplicity of code terms organized in a hierarchical form. Retrieval of data is accomplished by querying with a query code made up of terms which appear in the address codes.

DRPR:
FIG. 1 depicts a sample tree-structured hierarchy.

DEPR:
Before discussing the details of the preferred embodiment, a simple example of a hierarchical structure is described. This example hierarchy is not the actual one chosen for use in the present invention, but is used for illustrative purposes.

DEPR:
Consider a data-base entry which contains a volume of textual information which relates to "Siamese cats" "Siamese cats" can be considered as belonging to a hierarchy, as shown in FIG. 1. "Siamese Cats" 8 belongs to the more general group "cats" 6, which may contain, in addition to "Siamese cats", "Burmese cats", "Persian cats" 10, "Alley cats" 12, etc. "Cats", in turn, belongs to the group "mamnmals" 4, which may contain, in addition to "cats", "dogs" 14, "rats" 16, etc. The position of "Siamese cats" in a hierarchy of this kind may be visualized by the tree-shaped structure shown in FIG. 1.

DEPR:
In FIG. 1 the term "animals" 2 is called the "root" of the tree, while the other terms occupy the branches of the tree. The root is considered the "highest" level of the tree, and one "descends" from the root down the branches. Thus, in FIG. 1 "animal" stands at the highest level of the hierarchy, which is also the broadest level, including, as it does, all of the lower levels of the hierarchy. That is to say, everything which exists in this hierarchy is an animal.

DEPR:
The process may be visualized by considering the tree of FIG. 1 as a group of data paths. We may consider a path starting at Siamese Cats 8 and ascending toward the root (Animals 2). All the data on that path relates to Siamese Cats. However, we may choose to enter the tree at different points along that path, depending upon what level of information we desire.

DEPR:
Now suppose that we wish to query the data base for "Siamese cats". A query code containing the expression "Siamese cats" will retrieve the thread desired. On the other hand, so will a query which contains the expression "Animals". Thus, the hierarchical address code allows us to query the database with as little or as much specificity as desired.

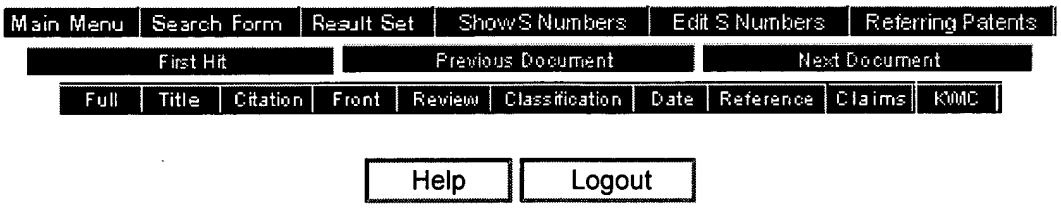
DEPR:
The first column, labeled "Attribute Entry", is an English-language description of the entry. The "level" column indicated the number of the hierarchical level of the entry. The "Query Code" contains the most concise REMDL code phrase which may be used to retrieve this item from a data thread. And the "Address Code" is the fully expanded REMDL description of the entry, indicating the most verbose description of the data entry possible.

DEPR:
Note that column 1 of the Bible of Microfiche Appendix A uses Tabs to indicate hierarchical level of the entry. For instance, line 5 contains ABC Sports, a level 1 entry. Line 20, "ABC Prime Time" is a level 2 entry, which is a "branch" of line 5, and is preceded by a single Tab. Line 21 is a branch of line 20, and therefore a level 3 entry, and preceded by a double Tab. And so on.

DEPL:
discussed above, this code appears in Microfiche Appendix A as line 265. Referring to line 265, column 1 contains the description of the entry, which in this case is "Book, Comic Book Format, Hardcover Print." Column 2 indicates that this entry occupies the highest level

of the hierarchy, level 1. Note, however, that this entry is a noun declension, which itself is a hierarchical form. Column 3 of line 265 is the query code, in this case /xD42B. This query code is the most concise query which may be used to retrieve the entry of line 265 when it appears in an address code, although more verbose forms may always be used.

DEPC:
USE OF A HIERARCHICAL ADDRESS CODE



5/11.00



Document Number 9

Entry 9 of 20 File: USPT Nov 24, 1998

US-PAT-NO: 5842212
DOCUMENT-IDENTIFIER: US 5842212 A

TITLE: Data modeling and computer access record memory
DATE-ISSUED: November 24, 1998

INVENTOR-INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY |
|---|---|---|---|---|
| Ballurio; Keith B. | Mansassas | VA | N/A | N/A |
| Edelstein; Matthew R. | Arlington | VA | N/A | N/A |
| Puckett; Brian B. | Oakton | VA | N/A | N/A |

ASSIGNEE INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY | TYPE CODE |
|---|---|---|---|---|---|
| Information Project Group Inc. | Herndon | VA | N/A | N/A | 02 |

APPL-NO: 8/ 610945
DATE FILED: March 5, 1996

INT-CL: [6] G06F 17/30
US-CL-ISSUED: 707/100; 707/1
US-CL-CURRENT: 707/100; 707/1
FIELD-OF-SEARCH: 707/1, 707/2, 707/101-102

REF-CITED:

U.S. PATENT DOCUMENTS

| PAT-NO | ISSUE-DATE | PATENTEE-NAME | US-CL |
|---|---|---|---|
| 5201046 | April 1993 | Goldberg et al. | 707/100 |
| 5257185 | October 1993 | Farley et al. | 707/100 |
| 5311438 | May 1994 | Sellers et al. | 364/468.02 |
| 5566333 | October 1996 | Olson et al. | 707/102 |
| 5590360 | December 1996 | Edwards | 707/102 |
| 5710917 | January 1998 | Musa et al. | 395/681 |

FOREIGN PATENT DOCUMENTS

| FOREIGN-PAT-NO | PUBN-DATE | COUNTRY |
|---|---|---|
| WO 90/08360 | July 1990 | WO |

ART-UNIT: 276

PRIMARY-EXAMINER: Kulik; Paul V.
ATTY-AGENT-FIRM: Fulbright & Jaworski LLP

ABSTRACT:

The system and apparatus for loading and retrieving information relates to a computer-implemented database management system for multiple source databases. The system also has a variety of database management tools. The system uses hierarchical, network, and relational structures to establish and maintain relationships between disparate categories of information in multiple records or databases within the system. Data entered into the system are stored in a common data repository in disk memory, which categorizes each source field independent from the source record definition. Separating the source record definition from the fields within each source record definition allows data to remain independent from the original structure of the information. Yet, source record definitions are maintained to show the relationships between data from different fields. The system allows all data to be referenced by any number of methods without regard to how the data was entered into the system. The system also allows any data modeling record (DMR) in the system to act as a menu, filter, or gateway to other DMRs or applications and to provide a data security system which gives a database manager sophisticated control of access to all DMRs and applications within the system. The system also provides for continuous modification of the database without any system down-time.

6 Claims, 9 Drawing figures

Document Number 9

Entry 9 of 20 File: USPT Nov 24, 1998

DOCUMENT-IDENTIFIER: US 5842212 A
TITLE: Data modeling and computer access record memory

ABPL:
The system and apparatus for loading and retrieving information relates to a computer-implemented database management system for multiple source databases. The system also has a variety of database management tools. The system uses hierarchical, network, and relational structures to establish and maintain relationships between disparate categories of information in multiple records or databases within the system. Data entered into the system are stored in a common data repository in disk memory, which categorizes each source field independent from the source record definition. Separating the source record definition from the fields within each source record definition allows data to remain independent from the original structure of the information. Yet, source record definitions are maintained to show the relationships between data from different fields. The system allows all data to be referenced by any number of methods without regard to how the data was entered into the system. The system also allows any data modeling record (DMR) in the system to act as a menu, filter, or gateway to other DMRs or applications and to provide a data security system which gives a database manager sophisticated control of access to all DMRs and applications within the system. The system also provides for continuous modification of the database without any system down-time.

BSPR:
This invention relates to an information management system, particularly a system suitable for developing and administering large and complex centralized or distributed databases. Unlike conventional database systems, the system and apparatus for loading and retrieving information does not store data records in static, structured tables or files. Instead, the system stores all data in a common data repository. The data stored in this repository are "independent" of the source records and the source fields used for data entry and display. The system allows hierarchical, network, and relational dynamic record alignment, which provides multiple database managers with the flexibility to structure data according to each manager's needs.

BSPR:
Conventional database systems link tables either as a hierarchical, a network, or a relational system. A hierarchical database structure uses one-way pointers to relate tables together in a fixed parent-to-child relationship. A hierarchical database constructs a one-to-many relationship between tables that looks like a tree. This hierarchical structure aids in understanding the relationship of a particular table with respect to other tables in the hierarchy, yet this rigid relationship limits the types of information that can be

available on any particular table in the hierarchy.

BSPR:
A network database structure is similar to a hierarchical structure in that tables are related in a fixed manner using pointers. A network structure, however, uses two-way pointers to create a many-to-many relationship between tables that looks more like a web than a tree. A network structure has duplexed relationships between tables, rather than the one-way parent-child relationships of a hierarchical structure. A network database system, however, can support a pure hierarchical structure as required.

BSPR:
Both the hierarchical and network database structures are based on a knowledge of the fixed location of tables. These structures are called navigational structures, which imply that a browsing user can understand the relationships between tables simply by traversing the structure. These navigational structures, however, sometimes suffer from the requirement of rigid pre-established relationships and the possibility of contamination of the one-way and two-way pointer chains. For the most part, hierarchical and network databases have been supplanted by relational databases, because relational databases do not require pre-planned relationships.

BSPR:
A relational database structure is not a navigational structure and does not have fixed pointers from one table to another. A relational structure consists of indices that are not limited to a hierarchical structure, but nevertheless relate records to each other. Relational databases operate using the principle of commonality between record formats to relate records to each other. For example, a "Name and Address" table might contain a "Customer Number." Associated tables, such as "Customer Orders," "Customers Handled," "Credit Profiles," and "Customer Complaints" might also include a "Customer Number" field, which would be used to associate records in the various tables.

BSPR:
Relational linkage structure enables records to be accessed and viewed from different perspectives, however, relational indices do not convey the relationship between individual records as well as hierarchical and network tables do. Also, hierarchical and network databases retrieve queries faster and use less computer processing power than relational databases, because the relationships in hierarchical and network databases are pre-established.

BSPR:
The system and apparatus for loading and retrieving information is a computer-implemented superstructure over an existing relational database management system. The system uses hierarchical, network, and relational structures to establish and maintain relationships between data, fields, and records in the system. Initially, the system uses hierarchical structures to order massive amounts of information possibly obtained from various disparate, distributed sources. Next, the system uses a network structure to cross-reference related data residing in different hierarchical structures. Finally, the system uses a relational structure to access the various hierarchical and network structures and report results of user queries.

BSPR:
An advantage of this invention is that it facilitates multiple management of and multiple access to disparate data in computer memory without concern for the source field and source record structure of the data. Another advantage of this invention is that it

provides a system that supports hierarchical, network, and relational record alignment. Other advantages of this invention are that it allows any record in the system to act as a menu, filter, or gateway to other records or applications and to provide a data security system that gives multiple information managers sophisticated control over access to data and applications within the system. Another advantage of the system is that it allows continuous modification of the data without computer system down-time.

DRPR:
FIG. 1 shows a preferred embodiment using hierarchical, network, and relational structures.

DEPR:
In the preferred embodiment, however, heavy processing by the RDBMS is avoided because the RDBMS is used primarily to handle only single format blocks, several indices, and various user interaction interfaces, usually through a display screen. Using a computer, the system directs the creation of the common data repository, the hierarchical linking of data modeling records (DMRs), the network linking of blocks within a DMR, the query engine, the intelligent DMR, and the rights matrix.

DEPR:
The system supports hierarchical, network, and the relational structures. FIG. 1 shows related DMRs linked using hierarchical, network, and relational structures according to the preferred embodiment. Each hierarchy 11, 12 includes a single root DMR 111, 121 and multiple children DMRs linked using one-way pointers. Pointers, both one-way and two-way, are stored as references to the linked item within DMR blocks, specifically within each DMR's data array. For example, hierarchy 11 may represent a mail-order database for a mail-order company with one general division and two specialty divisions. Such a database may contain customer "Name and Address" DMR 111, customer "Credit Profiles" DMR 112, "Customer Orders--General" DMR 113, "Customer Complaints" DMR 114, "Customer Orders--Furniture" DMR 115, and "Customer Orders--Automotive" DMR 116. Hierarchy 12 may be a human resources database for the employees of the mail order company with employee "Name and Address" DMR 121, "Customers Handled--General" DMR 122, employee "Timekeeping and Salary" DMR 123, employee "Division Roster--General" DMR 124, "Customers Handled--Automotive" DMR 125, "Customers Handled--Furniture" DMR 126, "Division Roster--Furniture" DMR 127, and employee "Division Roster--Automotive" DMR 128. Note that in these hierarchical structures a user may browse from one DMR to the next only in the direction of the one-way arrows.

DEPR:
FIG. 1 also shows relational structure 13 with a single-key alphabetical index. The preferred embodiment uses a unified single-key alphabetical index into which every common repository DMR is placed. This index is held in the traditional RDBMS that is incorporated "under" the system and stored on a server's storage devices. Each key in the relational structure 13 has a one-to-one relationship with every DMR in each hierarchy 11, 12. This relational structure can be used to access quickly the various hierarchical and network structures. Note that the various hierarchical structures may be products of not only different divisions within a single company, but they may also be products of completely different companies.

DEPR:
The preferred embodiment is based upon the use of DMRs, rather than tables, as basic building units for hierarchical structures. Instead of a table structure dictating exactly what fields may be used in a particular database or data structure, data structure rules developed

by individual managers control the development of a hierarchy. Data structure rules may allow only certain types of Record Types to be hierarchical children of a certain DMR. Information may be entered in a number of different ways depending on the number of Record Types a manager has created. Part of the root block content is the Record Type ID. All associated DMR blocks must contain the same Record Type ID. All DMR block information is held within the flat file on the services storage devices.

DEPR:
In addition to the system fields listed above, a root block contains two unique additional system fields that relate each DMR to a hierarchy: Root Record Identifier and Aspect. According to a preferred embodiment, each DMR has a place in one hierarchical database structure having a root DMR. The Root Record Identifier contains the Record Identifier of the root DMR in a DMR's hierarchical structure. The Aspect contains a string of Record Identifiers tracing the hierarchy of the DMR. The Root Record Identifier in this example is "111," which is the Record Identifier for the DMR with the Term "Name and Address" in the customer hierarchy 11 shown in FIG. 1. The Aspect, "111:113," represents that the DMR with the Record Identifier "113" is the parent of the instant DMR, and that the DMR with the Record Identifier "111" is the grandparent of the instant DMR. Every Aspect lists all of the parents of the instant DMR, from the direct parent to the root DMR of the hierarchy. Thus, the system fields indicate that the DMR shown is in a hierarchical structure of data for mail-order customers related to each other through customer order classification. The Root Record Identifier and the Aspect are used mainly to maintain the hierarchical referential integrity of the system.

DEPR:
FIGS. 3 and 4 show examples of disparate source structures. Source table A represents manufacturer orders of automobile parts. The fields available in source Table A are: dealer number, date of order, customer name, year of car, model of car, item number, quantity, and warranty information. Source table B represents automobile replacement parts orders. The available fields in Source table B are: name, address, make, model, catalog number, description, color, quantity, price per item, subtotal, tax, shipping, total, payment type, and account number.

DEPR:
FIG. 8 shows a rights matrix. The preferred embodiment rights matrix is a Progress RDBMS table used at the system level for subsystem control. Another aspect of the preferred embodiment is multiple control domains within the common data repository. Most databases offer two levels of basic security (user vs. nonuser & user vs. superuser). The preferred embodiment provides an additional level of security through individual subsystem operational control. A rights matrix controls access and use of the subsystems and the discrete functions encompassed in those subsystems. FIG. 8 shows eight levels of security, with level 8 being analogous to a superuser. The preferred embodiment can support any number of security levels with preferably the highest security level having access to all subsystems. Within a subsystem, a manager at level 6 may use the functions for creating, deleting, and modifying DMRs as well as creating, deleting, and modifying fields. In the rights matrix, each subsystem as well as its underlying functions appears as a discrete element. This enables a manager to decide exactly which rights are assigned to certain user groups on a function-by-function basis. Every request to access a function is matched against the rights matrix before the function can be performed. User level and subsystem requested are looked up in the rights matrix table for permission.

DEPR:
FIG. 9 shows a schematic for an intelligent DMR command executor. Intelligent commands (triggers) are held in the system DMR Chain-4 data arrays. If a user processes a DMR with Chain-4 array content that performs an action, then that action will "trigger." If a user trips a DMR trigger, the Record Identifier of the DMR is sent to the script language processor 81, preferably a 4GL program on file, which starts the operation of the resident intelligence in the DMR Chain-4 arrays. The various triggers include: requesting to read data in a DMR, traversing through a DMR to see hierarchical children DMRs, traversing back through a DMR to see hierarchical parent DMRs, using a DMR as a menu, requesting to trigger the DMR directly, and editing the DMR.

DEPR:
Thus, the system may develop, integrate, and administer large and complex databases using hierarchical, network, and relational structures. This system may, of course, be carried out in specific ways other than those set forth here without departing from the spirit and essential characteristics of the invention. Therefore, the presented embodiments should be considered in all respects as illustrative and not restrictive and all modifications falling within the meaning and equivalency range of the appended claims are intended to be embraced therein.

5/11.



Document Number 19

Entry 19 of 20 File: USPT May 27, 1986

US-PAT-NO: 4591983
DOCUMENT-IDENTIFIER: US 4591983 A

TITLE: Hierarchical knowledge system
DATE-ISSUED: May 27, 1986

INVENTOR-INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY |
|---|---|---|---|---|
| Bennett; James S. | Palo Alto | CA | N/A | N/A |
| Lark; Jay S. | Palo Alto | CA | N/A | N/A |

ASSIGNEE INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY | TYPE CODE |
|---|---|---|---|---|---|
| Teknowledge, Inc. | Palo Alto | CA | N/A | N/A | 02 |

APPL-NO: 6/ 628817
DATE FILED: July 9, 1984

INT-CL: [4] G06F 15/24
US-CL-ISSUED: 364/403; 364/468, 364/478, 235/385, 29/703
US-CL-CURRENT: 706/53; 235/385, 29/703, 700/103, 700/104, 705/29, 706/59, 706/904
FIELD-OF-SEARCH: 364/400-401, 364/403, 364/468-469, 364/478, 364/513, 364/518, 209/1-2, 209/546, 209/552, 235/385, 29/33K, 29/33R, 29/4R, 29/4M, 29/428-431, 29/469, 29/564, 29/568, 29/700-703, 29/711

REF-CITED:

U.S. PATENT DOCUMENTS

| PAT-NO | ISSUE-DATE | PATENTEE-NAME | US-CL |
|---|---|---|---|
| 4203204 | May 1980 | Murphy | 29/703 |
| 4310964 | January 1982 | Murphy | 29/469 |
| 4332012 | May 1982 | Sekine et al. | 364/468 |
| 4472783 | September 1984 | Johnstone et al. | N/A |
| 4484289 | November 1984 | Hemond | 364/478 |
| 4504919 | March 1985 | Fujii et al. | 364/478 |
| 4509123 | April 1985 | Vereen | N/A |

OTHER PUBLICATIONS

James Bennett et al., "SACON: A Knowledge-Based Consultant for Structured Analysis," Stanford University Rep. STAN-CS-78-688 (Sep. 1978).

Bennett & Engelmore, "SACON: A Knowledge-Based Consultant for Structured Analysis," Proc. of the Sixth Int. Joint Conf. on Artificial Intelligence, Tokyo (Aug. 20-23, 1979) pp. 47-49.
John McDermott, "R1: A Rule-Based Configurer of Computer Systems," Carnegie-Mellon Univ., Rep. CMU-CS-80-119 (Apr. 1980).
W. van Melle et al., The Emycin Manual, Stanford University, Rep. STAN-CS-81-855 (Oct. 1981).
Barr & Feigenbaum (eds.), The Handbook of Artificial Intelligence, William Kaufmann, Inc. (1982) vol. II, pp. 79-86, 150-154, vol. III, pp. 515-530, 541-556.

ART-UNIT: 236
PRIMARY-EXAMINER: Harkcom; Gary V.
ATTY-AGENT-FIRM: Leydig, Voit & Mayer

ABSTRACT:

A knowledge system has a hierarchical knowledge base comprising a functional decomposition of a set of elements into subsets over a plurality of hierarchical levels, a plurality of predefined functions or conditions of the elements within the subsets of a plurality of the hierarchical levels, and a predefined set of operations to perform on a user-defined set of elements responsive to the functional knowledge base. Preferably, the knowledge base is defined declaratively by assigning parent sets to offspring subsets to define the hierarchy, by indicating the conditions of the subsets which satisfy the predefined functions and by writing task blocks in an imperative language defining the sequence of operations to perform on the user-defined set of elements. Preferably the operations include matching, configuring and expanding the user-defined set of elements into the defined subsets of individual elements and evaluating the predefined functions, and the operations are executed recursively. In a specific embodiment the elements are available components for a system or item of manufacture, and the subsets of elements are sub-assemblies or functionally related components. The predefined functions define condition-action constraints to insure that the sub-assemblies have compatible components. Such a knowledge system has general applicability, is easy to maintain and incrementally modify, has transparent representation of the functional decomposition and the configuration operations, and provides explanation for an assessment of the configuration.

70 Claims, 13 Drawing figures

Document Number 19

Entry 19 of 20 File: USPT May 27, 1986

DOCUMENT-IDENTIFIER: US 4591983 A
TITLE: Hierarchical knowledge system

ABPL:
A knowledge system has a hierarchical knowledge base comprising a functional decomposition of a set of elements into subsets over a plurality of hierarchical levels, a plurality of predefined functions or conditions of the elements within the subsets of a plurality of the hierarchical levels, and a predefined set of operations to perform on a user-defined set of elements responsive to the functional knowledge base. Preferably, the knowledge base is defined declaratively by assigning parent sets to offspring subsets to define the hierarchy, by indicating the conditions of the subsets which satisfy the predefined functions and by writing task blocks in an imperative language defining the sequence of operations to perform on the user-defined set of elements. Preferably the operations include matching, configuring and expanding the user-defined set of elements into the defined subsets of individual elements and evaluating the predefined functions, and the operations are executed recursively. In a specific embodiment the elements are available components for a system or item of manufacture, and the subsets of elements are sub-assemblies or functionally related components. The predefined functions define condition-action constraints to insure that the sub-assemblies have compatible components. Such a knowledge system has general applicability, is easy to maintain and incrementally modify, has transparent representation of the functional decomposition and the configuration operations, and provides explanation for an assessment of the configuration.

BSPR:
EMYCIN is specifically designed as a domain-independent system for constructing rule-based consultant expert system programs. Domain knowledge is represented in EMYCIN systems primarily as condition-action production rules which are applied according to a goal-directed backward-chaining control procedure. Rules and consultation data are permitted to have associated measures of certainty, and incomplete data entry is allowed. The EMYCIN system includes an explanation facility displaying the line of reasoning followed by the consultation program, and answers questions from the client about the content of its knowledge base. To aid the system designer in producing a knowledge base for a specific domain, EMYCIN provides a terse and stylized language for writing rules; extensive checks to catch common user errors, such as misspellings; and methods for handling all necessary indexing chores.

BSPR:
In addition to production rules, the knowledge base for an EMYCIN system includes a hierarchical structure called a "context tree." The elemental representation of an object or idea is defined as a

context-parameter-value triple. The context refers generally to an instance of a particular context type, the parameter refers to an attribute of the context instance and the value refers to the particular value of the parameter for the particular context instance. The context tree is defined by parent and offspring declarations for the context types.

BSPR:
The instantiation of contexts is similar to the invocation of a subroutine for each context, the subroutine in effect being defined by various declarations in the context definition. A consultation is started by instantiating a root context and the branches from this root context define major steps in the consultation during which the offspring contexts of the root node are instantiated. Thus, the context definitions are used to structure the data or evidence required to advise a user about the root context. Besides consultation control, the context tree may be used to organize the distinguished components of some object, or for representing distinguished events or situations that happen to an object.

BSPR:
A version of XCON called R1 is described in John McDermott, R1: A Rule-Based Configurer Of Computer Systems, Department of Computer Science, Carnegie-Mellon University, (April, 1980). It is said that R1 has sufficient knowledge of the configuration domain and of the peculiarities of the various configuration constraints that at each step in the configuration process, R1 simply recognizes what to do. Consequently, it is said that little search is required in order for R1 to configure a computer system.

BSPR:
Although the XCON rule-based system represents a major improvement over traditional computer techniques in the field of computer configuration, it is rather difficult to change the constraint rules in XCON. In particular, to change a constraint rule, one needs to know what occurs before and after the constraint rule is applied. In general, the constraint rules are interlaced and spread, and they are not readily accessible for maintenance since in general they are not conceptually hierarchical.

BSPR:
The primary object of the invention is to provide a knowledge system for generalized representation and processing of hierarchical assemblies that is easily maintainable and extensible.

BSPR:
Another object of the invention is to provide an intelligible knowledge base representation for hierarchical assemblies and their functionality.

BSPR:
Briefly, in accordance with the broadest aspect of the invention, a knowledge system for generalized representation and processing of hierarchical assemblies has a hierarchical knowledge base comprising a decomposition of a set of elements into subsets over a plurality of hierarchical levels, a plurality of respective predefined functions or conditions of the elements within the subsets at a plurality of the hierarchical levels, and a predefined set of operations to perform on a user-defined set of elements responsive to the knowledge base. For ease of maintenance and extensibility, the knowledge base is defined declaratively by assigning parent sets to offspring subsets to define the hierarchy, by indicating the conditions of the subsets which satisfy the predefined functions, and by writing task blocks in an imperative language defining the sequence of operations to perform on the user-defined set of elements. Preferably, the

operations include operations for matching, configuring and expanding the user-defined set of elements into the defined subsets of individual elements and for evaluating the predefined functions, and the operations are executed recursively.

BSPR:
In accordance with a preferred embodiment of the invention, a knowledge-based configuration system has separate portions encoding a configuration checking strategy, a description of hierarchical functions of the product to be configured, a catalog of the parts and components which implement those functions, assembly constraints that check whether a given set of components will successfully implement their respective hierarchical functions. The configuration system applies the assembly constraints during configuration checking and if necessary, warns the user or modifies the given set of components to insure compliance with the assembly constraints.

DRPR:
FIG. 3 is a tree diagram of the functional hierarchy for the M1234 computer system shown in FIG. 2;

DRPR:
FIG. 4 is a tree diagram of a functional hierarchy for a standard mini computer;

DRPR:
FIG. 6 is a diagram of a bin tree corresponding to the functional hierarchy of FIG. 4 for the mini computer, including the final configuration of parts in the bins for the working example of Appendix III (A);

DEPR:
Turning now to FIG. 1, there is shown a block diagram generally designated 10 of a knowledge system for processing an initial configuration of elements or an order 11 to arrive at a final configuration or production request 12. The system 10 is recognized as a knowledge system since it has a domain-independent knowledge base interpreter 13 executing a built-in control procedure for interpreting a domain-dependent knowledge base 14. The knowledge base 14 encodes a generic configuration in a highly structured and transparent format. In general terms, the knowledge base interpreter 13 matches the elements of the initial configuration in the order 11 to the generic configuration in the knowledge base 14 to structure or configure the elements according to the generic configuration. The elements structured in terms of the generic configuration are stored in a working configuration memory 15.

DEPR:
Once the elements or parts of the order 11 are structured according to the generic configuration, the knowledge base intepreter 13 may apply functions or constraints 16 stored in the knowledge base 14 to the working configuration in the memory 15. The constraints 16, for example, indicate particular aspects of the working configuration 15 that are not apparent from the initial configuration or order 11. The constraints 16 may also indicate desirable changes to make to the working configuration 15. The knowledge base interpreter 13 may execute these indicated changes in order to change the working configuration. At the end of processing by the knowledge base interpreter 13, these executed changes are reflected in the final configuration 12.

DEPR:
In addition to providing a framework for clearly intelligible and maintainable representation of knowledge about a product, the functional hierarchy 22 provides the structure or framework for the

control procedure executed by the knowledge base interpreter 13 to match the parts in the order 11 to the generic configuration in the knowledge base 14 and to apply the constraints 16 to validate the order 11 and generate a production request 12. The working configuration memory 15 is organized into respective bins of parts 26 corresponding to the major functional components of the product defined by the functional hierarchy 22. Each bin of parts 26 corresponds to a structural or functional assembly in the product and the matching process is performed by inputting the initial configuration or list of parts 11 into an initial bin of parts 26 and sequentially matching the parts in parent bins to assembly descriptions for offspring bins and transferring the matching parts to the offspring bins. The task blocks 23 specify the particular assembly descriptions and the specific sequence of matching and transferring the parts. Before matching and transfering the parts from certain bins, however, the expansion rules 25 must sometimes be applied to "break open" certain packages of parts in the order 11 since different parts from the same packages sometimes match the assembly descriptions of different offspring bins. At the end of the transfer process, the bins of parts 26 contain respective parts from the initial configuration 11. Hence, the parts in the initial configuration 11 have been structured or configured according to the generic product configuration or functional hierarchy 22 and thus the constraints 16 may be applied to their respective bins of parts 26.

DEPR:
From the block diagram in FIG. 2 and other technical information about the M1234 computer, the knowledge engineer 20 generates a functional hierarchy description of the M1234 computer as illustrated in FIG. 3. In FIG. 3 the hierarchy is arranged in the form of a tree diagram generally designated 40, having an initial or root node 41, intermediate or branch nodes 42-46, and several terminal or leaf nodes 47-60. Each node in the functional hierarchy 40 represents a single functional component. The initial node 41 and the intermediate nodes 42-46 represent functional components that are composed of a number of more specific functional components. Specifically, the functional hierarchy 40 is defined by including declarations of parent functional components in the definitions of the intermediate 42-46 and terminal 47-60 functional components. The functional component MAINFRAME 42, for example, has the parent functional component SYSTEM 41. The functional hierarchy 22 of the knowledge base 14 (FIG. 1) corresponding to the tree diagram 40 in FIG. 3 is listed in Appendix II (A). According to the particular knowledge base syntax of Appendix II (A), it is said that the declared offspring functional component "COMPOSES" its parent functional component.

DEPR:
An order for a MINI-1000 minicomputer must at least include a set of parts, but the set may be empty. In general, an order for a product consists of three types of information. The first type is called "order information" consists of such information as the customer name and an order number. It is primarily used to identify the order and does not effect the configuration process. The second type called "order lines" is a list of parts ordered and the quantity of each part desired. In general the order lines include the product model number and additional or optional features requested by the customer. The list of parts ordered and the quantity of each part desired is checked by the knowledge system 10 to ensure compatibily. The third type called "field service parts" is a list of additional part numbers and are merely appended or said to be shipped "on top" of the order. The field service parts are typically replacement or spare parts for the customer. The field service parts are not intended to be built into the system and are not checked either for compatibility among themselves or with the rest of the order.

DEPR:
Once the order is received by the knowledge system 10, the parts in the order are checked for consistency by applying the constraints in the knowledge base (Appendix IV (D)). In general, constraints specify engineering or marketing conditions which must be satisfied for the order lines. The constraints also specify the particular action that should be taken in response to whether the conditions are satisfied when the constraints are applied. For the MINI-1000 minicomputer, the order must include at least one RAM-1000 random access memory board, at least one ROM-1000 read only memory board, at least one TERMINAL-1000 computer terminal, and at least one IF-1000 interface board. The action to take when any of these parts is found to be missing is to add one of the respective missing parts, since these parts are essential to the operation of the minicomputer. The minicomputer also has a number of essential components that must be added if missing, but which can only appear with a quantity of one. The minicomputer must have one and only one power supply (SUPPLY-05, SUPPLY-10, or SUPPLY-15), one and only one CPU-1000 central processing unit card, and one and only one CAGE-1000 card cage.

DEPR:
By using the techniques already described, the components and functionality of a complex product such as a computer can be transparently and explicity represented in the knowledge base. In accordance with another aspect of the present invention, the definition of configuration constraints is clearly separated from the configuration strategies and actions. In particular, a built-in control procedure is provided which permits the constraints to be defined for particular functional components and separately determines which parts are applicable to the respective constraints. At the most basic level, this separation requires a process which builds an explicit representation of the assemblies of the product from the parts in the order. Shown in FIG. 6, for example, is an explicit representation generally designated 90 for the MINI-1000 minicomputer and the example order given above. The explicit representation 90 is a tree of bins corresponding to the functional hierarchy of FIG. 4. In FIG. 6 the name of each bin corresponds to the respective functional component or bin class in the hierarchy of FIG. 4. Moreover, the particular instance of the class is identified by a numeral enclosed in parentheses. Although the examples in the appendices only use a single instance of each functional class, the knowledge engineer may write task blocks which create a specified number of bins for a specified functional class.

DEPR:
Due to the complexity introduced by packages and modification constraints, it is important that the components of the order are properly associated with corresponding constraints. Preferably, this correspondence is obtained by matching the components in the order to predefined assembly descriptions and then applying the constraints to the components which match the respective assembly descriptions. Also, the preferred method of performing the matching is by transferring or sorting the entire order through the bin tree, starting at the initial bin until the elemental parts end up in the terminal bins. For the MINI-1000 order processed in Appendix III (A), the result of the sorting is illustrated in FIG. 6.

DEPR:
The order in which parts are matched to assembly descriptions, sorted and checked by the constraints is all controlled by the steps in the task blocks. The execution of the steps in the task blocks is started by an executive program 100 shown in FIG. 7. In the first step 101, certain program variables are cleared. By clearing these program variables, space in the working memory is cleared so that a new bin tree may be built. Also, the trace memory is cleared in order to

record a new trace. In step 102 an initial bin is created and is given the name "UNASSIGNED". The name "UNASSIGNED" is given denoting that the particular product being configured is not yet known since the order has not yet been received by the knowledge system.

DEPR:
Translated into English, the UNASSIGNED task block above first inputs the order of parts. Then it determines the current date, the ordered parts or order lines of the order, and the product line corresponding to the ordered parts. It is presumed that the knowledge system is currently checking the order. Thus, the task block for the ordered product line is fetched from the knowledge base. This task block is, for example, the MINICOMPUTER task block in Appendix IVB. Then a FIND operation is performed to obtain a list of all the parts in the order lines, and the CONFIGURE operation is performed to create the initial bin in the bin tree, to transfer all the parts in the order lines into the initial bin, and to temporarily transfer execution to the MINICOMPUTER task block. When execution returns, the parts are configured in the bins as shown in the bin tree 90 of FIG. 6. A final set of EMPTY.BIN constraints are applied to ensure that all of the order lines in the order have been properly configured. If so, the FINAL.PARTS are determined and a production request is generated by the OUTPUT.ORDER statement.

DEPR:
The fact that a stack was used in the CONFIGURE subroutine 130 suggests that the CONFIGURE subroutine is advantageously used in a recursive fashion. In general, each task block associated with an intermediate node in the functional class hierarchy will have CONFIGURE operations in its associated task block for creating offspring bins preceded by a FIND operation to specify the parts to transfer to the offspring bins. This ensures that the parts initially placed into the root bin or added by product expansions become sorted through the bin tree to the terminal or leaf bins. Once the parts have filtered down to the terminal leaf bins, the most basic and elementary constraints may be applied to determine, for example, if specified parts or a specified number of parts are included in the terminal bins. To apply these constraints, a CHECK operation is provided to apply a specified subset of the constraints applicable to the functional class of the current bin.

DEPR:
The function (COUNT.PARTS part search) returns the number of (parts) in the current bin. If (search) is true (T), the function COUNT.PARTS will search the bin tree below and including the current bin returning the total number of (parts) in these bins.

DEPR:
The function (ORDERED? parts search) returns the value true (T) if there is at least one instance of any of the (parts) in the current bin. If the value of (search) is true (T), ORDERED? will search the bin tree below (and including) the current bin, returning true (T) if it finds instances of (parts) in these bins. The function (SYSTEM system-name1 system-name2 . . . ) tests whether the system type or product model of the order is one of the (system-names) indicated in the argument list. Each (system-name) should be a FUNCTION.CLASS having a PRODUCT.GEN declaration including SYSTEM.

DEPR:
In the knowledge base, the BIN.VAR declarations must include a LEGAL.VALUE field indicating the legal value for the bin variable (e.g., integer, string, etc.), a HOW.TO.DETERMINE field including a list of KBFUNCTIONs that can be used to determine the value of the bin variable, and a BIN.TYPES field including the bin types or function classes with which the variable is associated. It should be

noted that values of BIN.VAR instances can be referenced by constraints applied by task blocks that are lower in the bin tree than where the BIN.VAR instance is created. Thus, the BIN.TYPES should associate the bin variable with bins high enough in the bin tree so that the bin variables can be referenced by all of the appropriate contraints.

DEPR:
The preferred embodiment of the user interface and explanation facility 19 has several commands to enable the user to obtain an explanation of the operation of the system 10. The user can select options for either obtaining an explanation as a configuration is being run, or after the completion of a configuration. As a configuration is being run, the user can select one or more of six options. The command SHOW-STEPS instructs the explanation facility 19 to print the task block steps as they are executed. The command SHOW-DETERMINATION causes the explanation facility to print the values of bin variables as they are determined. The command SHOW-EXPANSIONS tells the explanation facility to print a description of the expansions applied to parts in the order. The command SHOW-PARTS causes the explanation facillity to indicate when parts are added to or deleted from a bin's contents. The command SHOW-FULL-TRACE or ALL causes the explanation facility to indicate all of the events described above.

DEPR:
In view of the above, a knowledge system has been described for generalized representation and processing of hierarchical assemblies. The configuration strategies and actions are defined in task blocks which are completely separate from the functional hierarchy which describes the generic configuration of the product and the parts catalog which defines the individual elements available for configuration. Moreover, the constraints are completely separate from the task blocks, functional hierarchy, and parts catalog so that the definition of configuration constraints are clearly separated from configuration checking strategies and actions. The knowledge base interpreter has a built-in control procedure and built-in functions which enable the knowledge engineer to design and implement readily a desired configuration checking strategy. Since the imperative language of the task blocks clearly defines the configuration checking strategies and actions, a trace of a configuration operation is easily stored in a trace memory including both the actual steps executed and the resulting actions such as warnings or modifications to the configuration. Thus, an explanation facility can be provided which generates an intelligible and comprehensible explanation of the configuration based on the record in the trace memory.

DEPU:
BIN TREE

DEPV:
A hierarchical structure of bins corresponding to the hierarchy of the functional classes of the bins.

DEPV:
(2) A numerical or logical value responsive to the domain of the function. A function of the parts in an assembly, for example, may indicate whether a specified number of specified parts are included in the assembly or whether the parts have specified conditions or attributes. In a computer program, functions are applied or evaluated at only specified instants of time.

DEPV:
The root of the bin tree

DEPV:
A list of parts ordered and the quantity of each part that will be checked.

CLPR:
51. The knowledge system as claimed in claim 50, wherein said functions include functions responsive to whether specified elements of the initial list are included within said respective lists of matching elements within the domains of the respective functions.

CLPR:
54. The knowledge system as claimed in claim 50 wherein said subsets correspond to hierarchical functions collectively performed by the combinations of their respective elements, and the definitions of said subsets explicitly state their respective functions.

CLPR:
56. The knowledge system as claimed in claim 50, wherein said declaration of hierarchical decomposition defines classes of subsets of said elements, and said control procedure includes configuring control steps for creating specified instances of said classes of subsets and adding corresponding elements of said initial list to the instances of said subsets.

CLPR:
58. The knowledge system as claimed in claim 57, wherein said set of control steps for recursively applying starts recursive application at the uppermost level of the hierarchical decomposition.

CLPR:
60. The knowledge system as claimed in claim 59, wherein said task blocks include imperative language statements including separate statements for creating and storing in said working configuration portion of memory the list of matching elements obtained by matching to a specified one of said subsets, and for applying a specified set of functions to the respective lists of matching elements within the domains of said functions.

CLPV:
a predefined declaration of hierarchical decomposition of said set of related elements into separately defined subsets of said elements, some of said separately defined subsets being composed of other of said separately defined subsets having fewer elements, and

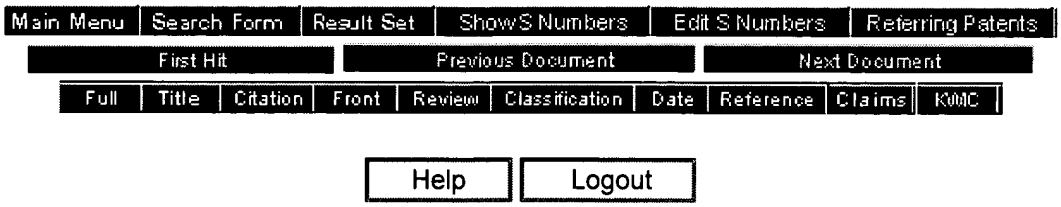
CLPV:
for said separately defined subsets of said elements, respective predefined functions of the respective elements in the subsets, the domain of each function thereby being its respective subset, such function having a predefined result obtained when the function is applied responsive to the specified elements within its domain,

CLPV:
successively applying said matching procedure to match said initial list to said subsets of elements composing said set of related elements, and for each of said subsets of elements composed of said subsets of fewer elements applying said matching procedure to match the previously obtained respective list of matching elements to each of said subsets of fewer elements composing said subset of elements to thereby obtain lists of matching elements representing the configuration of said initial list according to said hierarchical decomposition, and

CLPV:
applying said functions to the respective lists of matching elements within the domains of the respective functions and obtained by

matching to the respective subsets, and

CLPV:
said computer including means for executing said control procedure to thereby obtain said matching lists of elements representing a configuration of said initial list according to said hierarchical decomposition, and for applying said functions to the respective matching lists within the domains of the respective functions.



5/11.07



## Document Number 6

Entry 6 of 20 File: USPT Feb 2, 1999

DOCUMENT-IDENTIFIER: US 5867649 A
TITLE: Dance/multitude concurrent computation

ABPL:
This invention computes by constructing a lattice of states. Every lattice of states corresponding to correct execution satisfies the temporal logic formula comprising a Definitive Axiomatic Notation for Concurrent Execution (DANCE) program. This invention integrates into a state-lattice computational model: a polymorphic, strong type system; visibility-limiting domains; first-order assertions; and logic for proving a program correctness. This invention includes special hardware means for elimination of cache coherency among other things, necessary to construct state-lattices concurrently. The model of computation for the DANCE language consisting of four, interrelated, logical systems describing state-lattices, types, domains, and assertions. A fifth logical system interrelates the other four systems allowing proofs of program correctness. The method of the present invention teaches programs as temporal formulas satisfied by lattices of states corresponding to correct execution. State-lattices that are short and bushy allow application of many processors simultaneously thus reducing execution time.

BSPR:
The challenge to those wishing to exact optimum performance from highly parallel processors is to coordinate activities in processing nodes that are doing different things. Modifications to special-purpose languages like LISP and PROLOG to incorporate futures and guards, respectively, are problematic, addressing restricted application domains of largely academic interest.

BSPR:
Mutual exclusion (MutEx) via message passing (i.e. for distributed systems, no shared state) is horribly inefficient. If the application needs MutEx often, performance will suffer. "An Algorithm for Mutual Exclusion in Computer Networks," Glenn Ricart & Ashok K. Agrawala, Communications of the ACM, Vol. 24, No. 1, January 1981, for a discussion of prior art O(N) request/reply algorithm, which really needs O(N.sup.2) messages when everyone wants the lock. "A.sqroot.N Algorithm for Mutual Exclusion in Decentralized Systems," Mamoru Maekawa, ACM Transactions on Computer Systems, Vol. 3, No. 2, May 1985, reduces messages to O(.check mark.N) with an elegant multidimensional geometry. Sanders' generalized MutEx algorithm in more convoluted terms in her article, "The Information Structure of Distributed Mutual Exclusion Algorithms," Beverly A. Sanders, ACM Transactions on Computer Systems, Vol. 5, No. 1, August 1987. She considers deadlock freedom (absence of possibility that computation ceases because every process is waiting to acquire a lock held by a different process), but the basic model which she proposes needs further augmentation to detect and recover from deadlock. Of course,

only a wordy explanation substitutes for correctness proof. Therefore the desired interference freedom provided by MutEx is not achieved. Raymond in "A Tree-Based Algorithm for Distributed Mutual Exclusion," ACM Transactions on Computer Systems, Vol. 7, No. 1, February 1989, further reduced the number of messages to O(logN) by increasing the latency, which could easily become a significant burden if processes need the lock briefly.

BSPR:
Barriers are an artifact of the popular yet problematic process paradigm. A static collection of sequential processes alternately work independently and wait for laggards. FIG. 3 depicts four processes (201) synchronized with barriers (204). Each process is actively computing (202) and then wait idly (203) until all other processes reach the barrier. The best known algorithms for barrier-synchronization take O(logP) time (with a large constant) for both message-passing and shared-memory models. This corresponds with the depth of a tree using either messages or flags.

BSPR:
Consider an ideally parallelizable application that requires synchronization. A single processor would require N steps (time, T.sub.8) to perform the application with L iterations of a single loop each requiring N/L steps. When parallelized, barrier synchronization is required at the end of each iteration taking logP steps. Real systems take many steps for each level of the synchronization tree. So the parallel time for each iteration, T.sub.i is N/(L*P)+logP and the whole application T.sub.p, is T.sub.i *L or N/P+L*logP steps. Speedup, S, is the ratio of sequential time to parallel time, T.sub.s /T.sub.p. Efficiency, E, is the speedup obtained divided by the number of processors used, S/P. In this case, E=T.sub.s /T.sub.p *P=N/(N+P*L*logP).

BSPR:
Since the height of the binary tree is O(log p) it takes O(log p) steps for an EREW PRAM to emulate a PRIORITY PRAM. Much of Fich's chapter is devoted to convoluted schemes by which one model simulates another without mentioning any applications that benefit from a more powerful and expensive model.

BSPR:
Consider a function whose domain is a Cartesian product, say .function.:A.sub.1 x . . . xA.sub.n .fwdarw.B. Then it is customary to drop one pair of parentheses when applying .function. to an element (a.sub.1, . . . , a.sub.n).di-elect cons.A.sub.1 x . . . xA.sub.n. We also say .function. if an n-ary function.

BSPR:
Consider a function whose domain and co-domain coincide, say .function.:A.fwdarw.A. An element a.di-elect cons.A is called a fixed point of .function. if .function.(a)=a.

BSPR:
A model is a triple (T, S, ) containing a set of data types T, a set of domains containing states S, and an interpretation giving meaning to every symbol used in the language. For now, consider the data type domain, T, to be the integers. A state is a function mapping variables to values in T. For n state s in S and a variable X, let s [[X]] denote X's value in state s. Each k-place function symbol f has an interpretation s[[f]] which is a function mapping k elements in T to a single value:

BSPL:
The set A is called the domain of .function. and the set B is the co-domain of .function..

BSPV:
Phase 2: By evaluating (in parallel) a binary tree of all processors that want to write the highest priority processor is found.

DRPR:
FIG. 42 is a diagram showing nesting of domains in proof of noninterference;

DEPR:
A mathematical model of computation is presented with the salient advantage that the computation may be performed by hundreds or thousands of microprocessors simultaneously and efficiently. Five mathematical systems interrelate in special ways. The five systems are: temporal logic, domains, type theory, assertions and proofs. Representing programs as special temporal logic formulae that are satisfied (executed) by lattices of state transitions form the core of this invention and provide the simultaneity and efficiency desired. Domains contain variable-value bindings that restrict visibility of variables allowing necessary proofs of non-interference, but without the locks, semaphores, or messages that hobble the prior art attempts. The type theory adapts an elegant, lambda-calculus-based, polymorphic, object-oriented type system for the temporal logic execution espoused here. Assertions in first-order predicate calculus define specifications and allowable intermediate states for programs. Since only proofs can assure program correctness, a proof system called BPL associates inference rules and/or weakest-precondition predicate transformers with each language production. A proof of soundness for BPL is presented; a logic is sound if it infers true fact from true facts.

DEPR:
The domains visible at a particular state are both restricted and explicit.

DEPR:
Each DANCE interval may be thought of as a non-empty lattice of states partially ordered by their occurrence in time. A state is the values of a set of variables at an instant in time. The set of variables visible by a particular state is its domain.

DEPR:
Moszkowski doesn't recognize such bifurcation. Rather he uses the more traditional arbitrary interleaving of states. Though he makes reference to parallel computing, his intervals are always sequences, it's just that several predicates must be true over the same sequence of states. In this example the states v, and w contain variables seen only by w.sub.1. Similarly the states t and u contain variables seen only by w.sub.2. This is the essential, paradigmatic difference in DANCE from other models. Thus the importance of local variables and limited scoping in DANCE provided by domains.

DEPR:
Temporal Domains

DEPR:
Temporal domains hold bindings between identifiers (variables) and values. All state variables occur in some domain and at most one domain. Domains hold not only changeable states, but complete packages which contain procedures and functions as well as state variables.

DEPR:
Domains are essential for proving noninterference--that no other processor is modifying the variables any particular processor is

currently using. Noninterference is crucial to showing that parallel activities can occur together correctly ["An Axiomatic Proof Technique for Parallel Programs," Susan Owicki and David Gries, Acta Informatica 6, Springer-Verlag 1976].

DEPR:
Domain Trees

DEPR:
All domains for a single parallel program running on a single Multitude computer (which is comprised of many processors) form a rooted, directed tree. Intervals being executed (causing state change) exist in some leaf domain. The temporal predicates (commands) that control the interval can use any variable-value bindings in any domain below it in the tree. The domain tree will dynamically expand an contract as the program is executed. A new domain is created when a block is entered that declares local variables and is deleted when the block is exited.

DEPR:
Domain names are formed by a string of identifiers separated by periods representing the path in the domain tree from the root to the leaf. The local variables immediately declared within a subprogram are contained in a domain whose name is the subprograms identifier prepended (with a period) with the domain name in which it was invoked. Within subprograms, blocks that declare new variables may be encountered; a new domain leaf is grown. Non-nested blocks are assigned numbers starting with 1, in the order in which they are encountered. Nested blocks grow new domain leaves. Automatically numbered blocks cannot be confused with either named domains or subprogram domains since DANCE identifiers cannot start with digits. However domain identifiers and subprogram identifiers may be the same. However, an identifier's position in the name make clear whether it's a domain name or subprogram name.

DEPR:
In the following, d, d1, d2, etc. stand for domains and N(d) stands for the name of domain d. Define the domain operator , by:

DEPR:
Domains are neighbors if they have some variables they both can see:

DEPR:
Remote domains have no common variables:

DEPR:
Remote domains are needed to represent programs that span machine boundaries, i.e. distributed processing.

DEPR:
FIG. 42 shows the domains for the example above. FIG. 42 is not a set diagram in the usual sense of a subset drawn inside its superset. The set of nameable and therefore usable variables for a particular domain are all the variables declared within the domain, union variables declared in surrounding domains. Domains are necessary for proof of noninterference. The easiest way to assure non-interference is by restricting access to critical variables. Semaphores protect variables by serializing access to critical sections. DANCE protects variables with domains. By relying on the fact that no other interval can "see" critical, local variables, noninterference is assured.

DEPR:
Distributed systems have domains that do not form a tree--no single domain exists within which all other domains are contained. No shared address space, no combinable operations, just messages, usually

point-to-point sans confirmation. Remote procedure call (RPC) works just fine for much of what one would like to do at disparate geographical locations. (Note: I explicitly reject message passing paradigms for parallel processing that have one machine out of many microprocessors, in a single location. Only geography demands distribution.)

DEPR:
However there are worthwhile applications that demand geographical distribution that cannot utilize nested RPCs such as databases. RPCs are great for computation where the duration or lifetime of the domain which contains all accessed variables coincides with the duration of the invoked procedure. Such applications abound, i.e. photorealistic, 3D, animated rendering.

DEPR:
1) Divide the database into a large number of domains which may reside on machines across the country.

DEPR:
2) Impose a total ordering on the domains.

DEPR:
3) Allow only one RPC to be active in a domain--queue other received RLPC invocations first-in-first-out.

DEPR:
4) Make database accesses that need to maintain consistency between values in different domains use an RPC chain in the order of the domains used.

DEPR:
An RPC chain is a finite number of nested RPC calls each of which can be on a different machine. During the invocation phase, values of the database domains of interest are "read." During the return phase, values of the database are "written." Therefore any mapping from previous database values to new values is possible, and provide some initiating transaction the answer to, and confirmation of, its query.

DEPR:
The total ordering of domains makes it work. Suppose a single transaction occurs. It will create a chain through the database complete its last RPC and eventually terminate. If a transaction is stalled because its deepest RPC is stalled in a queue at a domain already in use, it will eventually be given access since the currently active RPC (and all RPCs ahead in the queue) will eventually terminate and release the domain (the inductive hypothesis). There is some "deepest" chain that occupies (has an active RPC chain) in the highest numbered domain. It will eventually terminate and release all its domains therefore possibly unblocking other RPC chains (the basis). Therefore all RPC chains will eventually terminate and total correctness can be proven.

DEPR:
This method only works efficiently when the transactions require domains evenly--each processor must know in which domain the value of interest lies. Tree-structured directories save much space and may work well for sequential accesses, but they cause "hot spots" that would substantially degrade the performance when distributed. On the other hand, if the number of domains were large enough so that the probability of any two transactions needing the same domain were small, the system would exhibit latency largely independent of the number of transactions--a blessing to travel agents worldwide.

DEPR:
By itself, a formal logical system is a mathematical abstraction--a collection of symbols and relations between them. A logical system becomes interesting when the formulas represent statements about some domain of discourse and the formulas are theorems that are true. This requires that we provide an interpretation to the formulas. An interpretation of a logical system maps each formula to true (T) or false (F). A logic is sound with respect to an interpretation if all its axioms and inference rules are sound. An axiom is sound if it maps to true; an inference rule is sound if its conclusion maps to true, assuming all the hypotheses are true. Thus, if a logical system is sound, all theorems in the logic are true statements about the domain of discourse. In this case, the interpretation is called a model for the logic.

DEPR:
Line 16 declares that the package contains a single item--the procedure "sort", with four parameters: "orig" of type "ar" to be supplied by the (caller (IN), "final" which holds the result (OUT), and "low" and "high" which indicate the range of "orig" to be sorted. The reserved word "SPREAD" causes the dynamic memory allocation of "orig" and "final" to put parts of it in many different memory banks (processors) so that "orig" and "final" may be concurrently accessed without contention. When concurrently accessible data structures are placed in a single memory bank, memory conflicts call significantly degrade performance. Line 17 demands as the precondition for invoking "sort" that the elements of the array are distinct--no duplicates. Modifying the code to handle duplicates requires an additional array to hold multiple matches of the chosen pivot. Lines 19 and 20, the postcondition, assure that when "sort" is done, the elements will be in sorted order. Other programs that invoke "sort" can use the postcondition in their own proof outlines. Such hierarchical proofs are essential for large programs. Line 23 concludes the package declaration for "Sort".

DEPR:
Line 33 begins the declaration of the procedure "sort" and must exactly match the declaration in the package "Sort". Lines 35 through 38 introduce local variables "inter" and "pivot" by existential quantification. They are only visible within the following begin-end block which corresponds to a domain when the program is executed.

DEPR:
Line 49 guards the third alternative in which there is more than one thing to sort. Line 51 invokes the "partition" procedure. Line 52 asserts that the data have been divided (partitioned) such that one part holds elements that are all less than the pivot and elements in the other part are greater than the pivot. This corresponds to the postcondition of "partition". Line 53, a simple semicolon, represents sequential composition of temporal logic formulae. Lines 54 and 57 recursively invoke "sort" on each part divided by "partition". Line 56, an ampersand, represents parallel composition; the two sorts may execute concurrently. Lines 55 and 58 assert that each part is in order which implies lines 63 and 64, the postcondition that the whole array (of this invocation) is in order. Lines 59 and 60 copy the value of the pivot concurrently with sorting of lines 54 and 57. Line 63 marks the end of the procedure "sort".

DEPR:
The fast Forrier transform (FFT) is a common signal processing operation used to extract a frequency spectrum from a sampled signal. The FFT can also be used to transform a filtered frequency domain representation to the time domain (Akl, p. 239).

DEPR:

proof of rule: The objective of this proof is given a host of assumptions that {P} is true for the first state of the interval and {Q} is true for the last state in the same interval. The model creates a host of little domains, each with its own parameter. The rule includes the requirement that each little domain be AccessOK, whereas the model does not. Execution attempts to construct a lattice that make the program true. If the model's subintervals j1, j2, . . . can be created that make each individual subinterval true, then the combined interval satisfies the whole parallel command. Each of the subintervals (j1,j2, . . . ) must be interference free in order to be constructed. So the requirement for interference freedom is implied in the model's definition. And now to the proof . . .

DEPR:
Unfortunately, older FORTRAN programs cannot be automatically converted into DANCE while still obtaining efficiencies possible. New, concurrently-accessible data structures that use combinable operations must be created. Fetch-add is useful for accessing arrays to assure that at most one interval is reading or writing any array element. Similarly, swap is useful for manipulating dynamic data structures like linked-lists and trees.

DEPR:
Define the domain operator , by:

DEPR:
The rules defining the scope of declarations and the rules defining which identifiers are "visible" at various points in the text of the program are described in this chapter. She formulation of these rules uses the notionl of a "declarative region." I have formally redefined such scoping rules using domains so that parallel correctness can be proven using Owicki-Gries. Such proofs formalize informal correctness justifications such as: "My interval, which is correct in isolation, is correct with an unknown collection of other intervals because no other interval can see or modify my state variables."

DEPR:
Though constructing state-lattices to satisfy temporal formulae is how the concurrency is accomplished, many other things must be done right to achieve efficiency. Logical systems for data types, domains, assertions, and correctness proofs must be integrated with the temporal logic. DANCE, programs will not run efficiently without the special features of the Multitude architecture. The operating system that allocates memory, sets and checks page and segment attributes, swaps pages for virtual memory, schedules and allocates work for processors, dynamically links and type-checks relocatable machine code, and handles input/output relys and supports on special features of the Multitude architcture, the DANCE language and its compiler.

DEPL:
which means "the domain of d is a set of pairs in which the first element, n, is a program identifier and the second element, v, is a value in the type of n." Types are sets of values. A domain contains the domains of all its ancestors in the domain tree. Ancestor domain names are formed by truncating their descendent's names:

DEPL:
where && is string concatenation. That is that domain d2 call "see" all the values in every domain d1 whose name is a shortened version of d2. All the values visible in domain d2 are either declared in the DANCE program text corresponding to d2, or are declared in some ancestor of d2.

DEPL:
where+is string CONcatenation. That is that domain d2 can "see" all

the values in every domain d1 whose name is a shortened version of d2, but there may be variables declared within a subprogram or block in d2 but not d1.

DEPV:
Lexical analysis.sup..fwdarw. Parsing.sup..fwdarw. Tree building.sup..fwdarw. Tree decoration.sup..fwdarw. Code generation

DEPV:
Variables declared within a subprogram or block are not visible outside. This corresponds to domains--variable/value bindings:

DETL:
_____ .di-elect cons. member of .epsilon slash. not memeber of .O slashed. empty set .OR right. subset .OR right. proper subset .OR left. superset .andgate. intersection .orgate. union .andgate..sub.i .di-elect cons..vertline. multiple intersection .orgate..sub.i .di-elect cons..vertline. multiple union x Cartesian product .smallcircle. composition .fwdarw. function space mapping, or implication double implication .function. function symbol .noteq. not equal .gtoreq. at least .ltoreq. at most .apprxeq. about equal .epsilon. empty sequence = equality by definition partial order irreflexive partial order .xi. infinite sequence .E-backward. there exists .A-inverted. for alI meaning .largecircle. next domain operator .diamond. sometime .SIGMA. summing quantifier .PI. multiplying quantifier infer

---

CLPR:
3. The method of claim 1 wherein visibility of said program variables in a particular state is restricted to domains.

ORPL:
A Tree-Based Algorithm for Distributed Mutal Exclusion, K. Raymond, ACM Transactions on Computer Systems, vol. 2, #1, Feb. 1989.

Document Number 6

Entry 6 of 20 File: USPT Feb 2, 1999

US-PAT-NO: 5867649
DOCUMENT-IDENTIFIER: US 5867649 A

TITLE: Dance/multitude concurrent computation
DATE-ISSUED: February 2, 1999

INVENTOR-INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY |
|---|---|---|---|---|
| Larson; Brian Ralph | Mpls. | MN | N/A | N/A |

ASSIGNEE INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY | TYPE CODE |
|---|---|---|---|---|---|
| Multitude Corporation | Bayport | MN | N/A | N/A | 02 |

APPL-NO: 8/ 589933
DATE FILED: January 23, 1996

INT-CL: [6] G06F 15/16
US-CL-ISSUED: 395/200.31
US-CL-CURRENT: 709/201
FIELD-OF-SEARCH: 364/DIG.1MSFile, 364/DIG.2MSFile, 395/377, 395/376, 395/379, 395/382, 395/385, 395/390, 395/391, 395/561, 395/800, 395/705, 395/706, 395/712, 395/200.31, 395/800.28, 395/800.3, 395/800.36, 340/825.8

REF-CITED:

U.S. PATENT DOCUMENTS

| PAT-NO | ISSUE-DATE | PATENTEE-NAME | US-CL |
|---|---|---|---|
| 4814978 | March 1989 | Dennis | 395/377 |
| 4833468 | May 1989 | Larson et al. | 340/825.8 |
| 5029080 | July 1991 | Otsuki | 395/377 |

OTHER PUBLICATIONS

Reference Manual for the ADA Programming Language, US Dept. of Defense, Jul. 1980.
Communicating Sequential Processes, C.A.R. Hoare, Comm. of the ACM, Aug. 1978, vol. 21, No. 8.
An Optimum Algorithm for Mutual Exclusion in Computer Networks, Glen Ricant et al. Comm. of the ACM, Jan. 1981, vol. 21, No. 1.
An Algorithm for Mutual Exclusion in Decentralized Systems, M. Maekawa, CAM Transactions on Computer Systems, vol. 3, No. 2, May 1985.
The Information Structure of Distributed Mutual Exclusion Algorithms,

Sanders, ACM Transactions on Computer Systems, vol. 5, No. 3, Aug. 1987.
A Tree-Based Algorithm for Distributed Mutal Exclusion, K. Raymond, ACM Transactions on Computer Systems, vol. 2, #1, Feb. 1989.
Algorithms for Scalable Synchronization on Shaved Memory Multiprocessors, J. Mollor-Crumley, ACM Transactions on Computer Systems, vol. 9, No. 1, Feb. 1991.
Verification of Sequential and Concurrent Programs, Apt. & Olderog; Springer-Verlag, 1991.
The Science of Programming, D. Gries Springer-Verlag, 1981.
Computer Architecture R. Karin Prentice-Hall (1989) (CPT. 5).
The Design and Analysis of Parallel Algorithms, S. Akl, Prentice-Hall, 1989.
Executing Temporal Logic Programs, B.C. Moszkowski, Cambridge University Press, Copyright 1986, pp. 1-125.

ART-UNIT: 235
PRIMARY-EXAMINER: Harrell; Robert B.
ATTY-AGENT-FIRM: Haugen and Nikolai, P.A.

ABSTRACT:

This invention computes by constructing a lattice of states. Every lattice of states corresponding to correct execution satisfies the temporal logic formula comprising a Definitive Axiomatic Notation for Concurrent Execution (DANCE) program. This invention integrates into a state-lattice computational model: a polymorphic, strong type system; visibility-limiting domains; first-order assertions; and logic for proving a program correctness. This invention includes special hardware means for elimination of cache coherency among other things, necessary to construct state-lattices concurrently. The model of computation for the DANCE language consisting of four, interrelated, logical systems describing state-lattices, types, domains, and assertions. A fifth logical system interrelates the other four systems allowing proofs of program correctness. The method of the present invention teaches programs as temporal formulas satisfied by lattices of states corresponding to correct execution. State-lattices that are short and bushy allow application of many processors simultaneously thus reducing execution time.

7 Claims, 45 Drawing figures

5/1●6



## Document Number 2

Entry 2 of 20 File: USPT Oct 26, 1999

US-PAT-NO: 5971589
DOCUMENT-IDENTIFIER: US 5971589 A

TITLE: Apparatus and method for managing and distributing design and manufacturing information throughout a sheet metal production facility
DATE-ISSUED: October 26, 1999

INVENTOR-INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY |
|---|---|---|---|---|
| Hazama; Kensuke | Yorba Linda | CA | N/A | N/A |
| Kask; Kalev | Irvine | CA | N/A | N/A |
| Sakai; Satoshi | Newport Coast | CA | N/A | N/A |
| Schwalb; Moshe | Irvine | CA | N/A | N/A |

ASSIGNEE INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY | TYPE CODE |
|---|---|---|---|---|---|
| Amadasoft America, Inc. | Lamirada | CA | N/A | N/A | 02 |

APPL-NO: 8/ 700671
DATE FILED: July 31, 1996
PARENT-CASE:
RELATED APPLICATION DATA This application claims the benefit of U.S. Provisional Application No. 60/016,958, filed May 6, 1996, entitled "Apparatus and Method for Managing and Distributing Design and Manufacturing Information Throughout a Sheet Metal Production Facility," the disclosure of which is expressly incorporated herein by reference in its entirety.

INT-CL: [6] G06F 19/00, G06T 17/00
US-CL-ISSUED: 364/472.01; 364/474.07, 364/474.24
US-CL-CURRENT: 700/145; 700/165, 700/182
FIELD-OF-SEARCH: 364/468.01, 364/468.03, 364/468.04, 364/468.24, 364/468.25, 364/472.01, 364/474.07, 364/474.24, 364/512, 345/419, 345/420, 345/158

REF-CITED:

U.S. PATENT DOCUMENTS

| PAT-NO | ISSUE-DATE | PATENTEE-NAME | US-CL |
|---|---|---|---|
| 4912644 | March 1990 | Aoyama et al. | 364/468.04 |
| 5029462 | July 1991 | Wakahara et al. | 72/379.2 |
| 5089970 | February 1992 | Lee et al. | 364/468 |
| 5115400 | May 1992 | Watanabe et al. | 364/474.24 |
| 5237647 | August 1993 | Roberts et al. | 395/119 |
| 5276606 | January 1994 | Mizukami et al. | 364/191 |
| 5297054 | March 1994 | Kienzle et al. | 364/474.24 |
| 5307282 | April 1994 | Conradson et al. | 364/468.04 |
| 5315522 | May 1994 | Kauffman et al. | 364/474.07 |
| 5396265 | March 1995 | Ulrich et al. | 345/158 |
| 5429682 | July 1995 | Harlow, Jr. et al. | 118/681 |
| 5434791 | July 1995 | Koko et al. | 364/468 |
| 5453933 | September 1995 | Wright et al. | 364/474.23 |
| 5485390 | January 1996 | Leclair et al. | 364/474.24 |

FOREIGN PATENT DOCUMENTS

| FOREIGN-PAT-NO | PUBN-DATE | COUNTRY |
|---|---|---|
| 0290809 | November 1988 | EP |
| 0397904 | November 1990 | EP |
| 0402475 | December 1990 | EP |
| 0419013 | March 1991 | EP |
| 0485766 | May 1992 | EP |
| 0664186 | July 1995 | EP |
| 1-309728 | December 1989 | JP |
| 1-309727 | December 1989 | JP |
| 2-15827 | January 1990 | JP |
| 2-15828 | January 1990 | JP |
| 5216525 | August 1993 | JP |
| 7121418 | December 1995 | JP |

OTHER PUBLICATIONS

An English Language Abstract of JP 5-216525, Aug. 1993.
An English Language Abstract of JP 2-15828, Jan. 1990.
An English Language Abstract of JP 1-309727, Dec. 1989.
An English Language Abstract of JP 63-154220 Jun. 1988.
An English Language Abstract of JP 2-15827, Jan. 1990.
An English Language Abstract of JP 1-309728. Dec. 1989.
Chu et al., Wesley W., Cooperative Query Answering Via Type Abstraction Heirarchy, Computer Science Department Technical Report, CSD-900032, Department of Computer Science, University of California, Los Angeles, pp. 1-28 (Oct. 1990).
Chiang, Kuorong, Automatic Generation of Type Abstraction Hierarchies For Cooperative Query Answering (A dissertation submitted as part of the requirements for the degree of Doctor of Philosophy in Computer Science), University of California, Los Angeles, pp. 1-98 (1995).
Mortenson, Michael E., Geometric Modeling (Title Page & Table of Contents), John Wiley & Sons,New York, (1988).
Foley et al., James D.The Systems Programming Series: Fundamentals of Interactive Computer Graphics, (Title Page & Table of Contents), Addison-Wesley Publishing Co., Reading, Massachusetts (Mar. 1983).
Mantyla, Martti, An Introduction To Solid Modeling (Title Page & Table of Contents), Computer Science Press, Inc. Rockville, Maryland (1988).

Wesley et al., M.A., "Fleshing Out Projections", IBM J. Res. Develop., vol. 25, No. 6, pp. 934-954 (Nov. 1981).
Aomura et al., Shigeru, "Creating Solid Model With Machine Drawings", the Sixth Computational Mechanics Conference, JSME, No. 930-71, pp. 497-498, Japan (1983).
Aomura, Shigeru, "Recent Trends And Future Prospect of Research And Practical Use (Automatic Reconstruction of 3D Solid From Drawings", JSME, No. 586-61, pp. 2180-2187, Japan (1995).
Open GL Reference Manual (Title Page & Table of Contents), Release 1,Open GL Architecture Review Board, Addison-Wesley Publishing Co., Reading, Massachusetts (Jan. 1995).
Open GL Programming Guide (Title Page & Table of Contents), Release 1,Open GL Architecture Review Board, Addison-Wesley Publishing Co., Reading, Massachusetts (Jun. 1995).
RenderWare, API Reference Manual (Title Page & Table of Contents), V2.0, Criterion Software Ltd., United Kingdom (Oct. 1995).
2D-3D: UNKEI/Solid and UNKEI/ Drawing Check & Projection/Reconstruction System, Sales Brochure, Toyo Engineering Corp. (TEC), Tokyo, Japan (1993).
Naessens, Diederik, "Flexible Automation on Press Brakes", American Machinist, pp. 36-39 (Jun. 1994).
Wysong Literature, The Perfect Forming Touch: New, PH Plus Series, DNC Press Brakes, Cat. PHP-1, Wysong & Miles Company, Greensboro, North Carolina (1993).
Bourne, David A., "Intelligent Manufacturing Workstrations", Knowledge-Based Automation of Processes, Session at the 1992 ASME Winter Annual Meeting (Nov. 1992).
Wang, Cheng-Hua, "A Parallel Design System For Sheet Metal Parts", Mechanical Engineering Report, presented to the Mechanical Engineering Department, Carnegie Mellon University, Pittsburagh, Pennsylvania, pp. 1-31 (May 1992).
Wang et al., Cheng-Hua, "Concurrent Product/Process Design With Multiple Representations Parts", IEEE, No. 1050-4729/93, pp. 298-304 (1993).
Amada Unfold: Manual For Autocad, Table of Contents, Index & pp. 1-27, U.S. Amada, Ltd., Buena Park, California (Mar. 1994).
Amada Unfold: Manual For Cadkey, Table of Contents, Index & pp. 1-18, U.S. Amada, Ltd., Buena Park California (May 1994).
Amada Windows Unfold: Manual For Cadkey, Table of Contents & pp. 1-13, U.S. Amada, Ltd., Buena Park, California (Nov. 1995).
AMACOM: AP40 Literature, Version 4, Amada Co., Ltd., Japan (Jul. 1996).
AMACOM: AP60 Literature, Amada Co., Ltd., Japan (Jul. 1996).
AMACOM: AP200 Literature, Amada Co., Ltd., Japan (Jul. 1996).
"Method For Understanding Drawing Attributes For 3D Models", IBM Technical Disclosure Bulletin, vol. 37, No. 07, pp. 99-104 (Jul. 1994).
Tseng et al., Yuan-Jye, "Recognizing Multiple Interpretations Of Interacting Machining Features", Computer-Aided Design, vol. 26, No. 9, pp. 667-688 (Sep. 1994).
Gu et al., P. "Product Modelling Using Step", Computer-Aided Design, vol. 27, No. 3, pp. 163-179 (Mar. 1995).
An International Search Report mailed Oct. 1, 1997 with Application No. PCT/US 97/07472.
An International Search Report mailed Nov. 6, 1997 with Application No. PCT/US97/07473.
An International Search Report mailed Nov. 4, 1997 with Application No. PCT/US97 07474.
An International Search Report mailed Oct. 1, 1997 with Application No. PCT/US 97/07471.
Amada Windows Unfold: Manual for Cadkey, Table of Contents, pp. 1-35, & Index, U.s. Amada, Ltd., Buena Park California (Nov. 1995).
Papanikolopoulos, Nikolaos P., "FORS: A System For Flexible Design," Conference Proceedings: 1990 IEEE International Conference On Systems, Man, And Cybernetics, Nov. 4-7, 1990, Los Angeles, California, pp.

724-726.
Patent Abstracts of Japan, vol. 18, No. 689 (P-1850), Dec. 26, 1994, JP 06-274219.
An International Search Report issued with PCT App. No. PCT/US 97/07474, 1998.
An International Written Opinion issued in counterpart PCT Application No. PCT/US97/07473, on Sep. 28, 1998.
An International Written Opinion issued in counterpart PCT Application No. PCT/US97/07472, on Oct. 16, 1998.
An International Preliminary Examination Report issued in counterpart PCT Application No. PCT/US97/07474, on Nov. 20, 1988.

ART-UNIT: 276
PRIMARY-EXAMINER: Grant; William
ASSISTANT-EXAMINER: Garland; Steven R.
ATTY-AGENT-FIRM: Greenblum & Bernstein P.L.C.

ABSTRACT:

A system and method are provided for developing a bend model of a part to be produced at an intelligent production facility. In accordance with an aspect of the disclosed system and method, faces of the part are detected based on initial part information, and bendlines of the part are identified based on the detected faces of the part. Further, additional part information is generated by performing a predetermined operation (e.g., a folding operation or an unfolding operation) on the detected faces of the part. The disclosed system and method also include other capabilities, such as the capability to perform clean-up operations on initial, 2-D part information and to selectively eliminate part thickness representations in order to facilitate the preparation of a 3-D representation of the part from the modified 2-D part information.

152 Claims, 95 Drawing figures

Document Number 2

Entry 2 of 20 File: USPT Oct 26, 1999

DOCUMENT-IDENTIFIER: US 5971589 A
TITLE: Apparatus and method for managing and distributing design and manufacturing information throughout a sheet metal production facility

BSPR:
According to yet another feature of the invention, the data relating to the initial part information may comprise part entity data representing, at least, line entities of the part, and the face detecting system may be adapted to perform a loop and entity analysis of the part based on the part entity data to detect the faces of the part. The loop and entity analysis may initially be performed on an outside boundary of the part and then performed on inner boundaries and areas of the part. The face detecting system may generate an initial linked list of entities when performing the loop and entity analysis on the outside boundary of the part, such that the initial linked list of entities define an outward loop and boundary of the part. The face detecting system may further generate an additional linked list of entities when performing the loop and entity analysis on the inside boundaries and areas of the part, such that the additional linked list of entities define inner loops and boundaries of the part. In addition, the face detecting system may further comprise a system for generating a loop tree based on the outward loop defined by the initial linked list of entities and the inner loops defined by the additional linked list of entities. Further, the face detecting system may detect the faces of the part based on the loop tree and the sequence of boundaries defined by the initial linked list entities and the additional linked list of entities.

DEPR:
The part information may comprise, for example, a part or order reference number, the customer's name, a brief description of the part, the batch size or quantity, and scheduled delivery date. The bend model data may include, for example, part geometry and manufacturing data, such as the overall dimensions of the part (e.g., width, height, depth), and part material information such as the material type (e.g., steel, stainless steel, or aluminum), thickness and tensile strength. Further, feature extraction data may be manually entered and/or automatically generated to identify the key features of the part and to facilitate similar part searches and other searches of the database. The feature extraction data may be stored in a separate data file in database 30, or may be stored with the bend model data and other job information for each part. The feature extraction data may comprise, for example, features of the part such as the number of surfaces or faces, the number or types of bends present (e.g., a positive bend between two faces or a negative bend between two faces), the relationships between the faces and/or the number of holes or other types of openings in the part. As discussed more fully below, such data may be represented and

organized in a feature based part matrix and/or a sequence of search keys (see, e.g., FIGS. 5-7 below). Lastly, bend line information may be entered at server module 32 for storage in database 30. The bend line information may comprise, for example, pertinent bend line information for each bend in the part, including the bend angle, the bend length, the inside radius (IR) of the bend, the amount of deduction, and the bend direction (e.g., front or back).

DEPR:
The manner in which the distance and feature/shape related to the search keys are modified during the cooperative search may be executed according to various methods and techniques. As described above, the amount by which to vary the distance may depend on the current value of the distance. The distance amount (e.g., 4 bendlines) may be modified to a distance range (e.g., 3-5 bendlines) to expand and make the search more cooperative. For the features or shapes, modification of the search keys may also be performed to identify similar parts. The features or shapes may be modified through a hierarchical structure of feature types. For example, the current feature type (e.g., a four bend box) may be modified to a less complex feature type (e.g., a three bend box) that is related and within the same feature type. The hierarchical structure by which the features/shapes are modified may be predetermined and developed based on different methodologies, such as type abstraction hierarchy (TAH). More information on TAH and TAH generation are provided, for example, in CHU et al., Wesley W., Cooperative Query Answering via Type Abstraction Hierarchy, CSD-900032, Department of Computer Science, University of California, Los Angeles, (October 1990) and CHIANG, Kuorong, Automatic Generation of Type Abstraction Hierarchies for Cooperative Query Answering, a dissertation submitted as part of the requirements for a Degree of Philosophy in Computer Science, University of California, Los Angeles, (1995), the disclosures of which are expressly incorporated herein by reference in their entireties.

DEPR:
After all of the inside loops have been defined (e.g., after all of the entities have been selected twice in the example of FIG. 10G), the resultant loops may be used to construct a loop tree.

DEPR:
FIG. 10H illustrates an exemplary loop tree that may be defined based the detected loops L1-L4. The outside loop (L4) of the part may be defined as the root of the tree, with each inside loop (L1-L3) that has a common entity with the outside loop being defined as children of the root. The presence of common entities may be detected based on analyzing and comparing the linked list of entities that define each loop. If additional entities (e.g., holes or openings) are detected within the inside loops, then these loops may be defined as children of the inside loops (i.e., grandchildren of the root of the loop tree) within which they are located.

DEPR:
After the face detection procedure has been performed at step S.124, a bendline detection operation may be performed at step S.126. As shown for example in FIG. 11A, when detecting and analyzing the loops of a part at step S.124, the face detection logic of the invention may utilize the loop tree to define the face information and store the detected faces as nodes in a bend graph data structure. The faces of the part may be detected from the sequence of the outside and inside loops in the loop tree. As indicated above, each of the loops may include a link list of entities or lines. These entities may be used to define the boundaries of each face of the part. The bendline detection operation of step S.126 may then be performed to determine the relationship between the faces and bendlines of the part. The

bendline detection operation of step S.126 may include bendline detection logic for detecting all of the bendlines between the various faces of the part by searching for common edges or line entities between any two adjacent faces. Further, for faces that connect at more than one area (e.g., when applying the bendline detection algorithm to a 3-D model--see, e.g., FIG. 12 discussed below) a number of heuristics may also be applied to detect and select the minimum number of bendlines for the part. The detected bendlines then may be stored as connecting agents between the face nodes to produce the final bend graph data structure, as shown for example in FIG. 11B.

DEPR:
After performing auto-trimming and cleanup functions on the 3-D drawing, at step S.144 a face detection operation may be performed to detect and define each of the faces of the sheet metal part. Face detection for the 3-D drawing may be performed by analyzing and detecting each of the faces in 2-D space and developing a loop tree, similar to that described above. Face detection may be executed by starting at any predetermined entity. For example, the left most entity (i.e., the entity with the lowest X-coordinate) may be used as the initial entity. Thereafter, a plane may be defined by taking the initial line entity and another connecting or adjacent line entity (e.g., any entity with a common endpoint to the initial entity). A face detection operation may then be performed using loop and entity analysis, such as that described above with respect to FIGS. 10A-10H. As each entity is detected within the defined 2-D plane, the various outside and inside loops may be defined and the entities may be marked (e.g., by setting or incrementing a flag of the selected entity) to indicate that they have been selected and included in a linked list defining one of the loops in that plane.

DEPR:
After all of the entities have been detected, the resulting loops may be used to develop a loop tree for each of the 2-D planes that were analyzed. As discussed above, a loop tree may be provided to determine the faces and opening or holes in the sheet metal part. For a 3-D drawing, a loop tree may be developed for each of the planes of the sheet metal part. The loops detected within each plane may be grouped and analyzed to develop each loop tree. The root of each tree may be defined as the outside loop detected in the plane, with each inside loop of that plane that has a common entity with the outside loop being defined as children of the root. The presence of common entities may be detected based on analyzing and comparing the linked list of entities that define each loop. If additional entities (e.g., holes or openings) are detected within the inside loops of the plane, then these loops may be defined as children of the inside loops (i.e., the grandchildren of the root of the loop tree) within which they are located. The generated loop trees may then be used to detect all of the faces of the 3-D drawing. The detected faces may then be stored as nodes in a bend graph data structure.

DEPR:
Finally, at step S.192, the inside loops, holes and shapes of the part may be detected in accordance with the teachings of the face detection procedure of the present invention. The various holes and shapes provided on the interior of the faces of each view may be detected by looping through the various lines and boundaries of the part from the exterior of the part towards the center. Loop and entity analysis may be performed on each view of the part in the 2-D drawing. By analyzing each view from the outside and working inwardly towards the center of the part, the detected loops will define the boundaries and areas of the material and openings of the part based upon a cyclic sequence (e.g., material, opening, material, etc.). A loop tree, such as that in FIG. 10H, may be developed for each view

to determine the location of the faces and any openings within each face. Unconnected entities, such as floating arcs or lines, within the faces of the part may also be detected and eliminated during step S.192.

DEPR:
After the bendlines have been identified and the mold lines have been added, the 3-D clean-up process may further process the 3-D representation of the part to further clean all bendlines and trim the faces of the part. Due to frequent ambiguities in the views of the 2-D, three view drawing data, superfluous sections of the faces may be generated in the 3-D representation of the part. The 3-D clean-up process may identify these superfluous sections of the faces and trims the faces using sheet-metal domain knowledge (e.g., knowledge relating to what is unfoldable). Other extraneous information, such as extra holes or openings, may also be identified and eliminated. As a result, the superfluous sections of the part may be removed and the 3-D representation may provide a more accurate representation of the sheet metal part. FIG. 51A illustrates an exemplary section of a part before cleaning the bendlines and trimming the faces, and FIG. 51B shows the section of the part after cleaning and trimming has been performed.

DEPR:
In addition to rendering each of the various view displays noted above, the bend model viewer view class may be implemented with other features. For example, the bend model viewer may include and maintain a selection set to indicate those items or entities in the current view that are selected or highlighted by the operator. In accordance with an aspect of the present invention, an operator may be permitted to select faces, bendlines and other features of the rendered part in order to modify the data relating to the selected items or to perform certain operations of those items of the part. For instance, an operator may be permitted to select a face of the displayed part and change the dimensional data of the face along its width or length. The operator could also be permitted to modify the various bend data associated with each bendline, such as the bend angle or V-width.

DEPR:
Various other features and embodiments may also be implemented in the present invention in order to aid in the design and manufacturing of components at the factory. For example, a bar code system may be implemented to keep track of and access information concerning each customer's order. A bar code with a predetermined reference or job number may be assigned to each part ordered by a customer. This bar code may be used for accessing and retrieving job information from database 30. A bar code reader or scanner, such as a Barcode Anything SCAN CCD sensor from Zebra Technologies VTI, Inc., Sandy, Utah, may be provided at each of the locations to permit users to scan the bar code for a particular job in to the server module or station module and to access and retrieve critical design and manufacturing information associated with that part that is stored in database 30. The bar code reader may be plugged into the computer of each of the station modules and/or the server module. The bar codes may be formatted in accordance with any conventional bar code formats, such as UPC-A, Codabar, Code 39, EAN/JAN-8 or Plessey, and the resultant bar code number may be translated in accordance with a lookup table to find the corresponding job reference number and/or file name in order to retrieve the job information from the database. Alternatively, the job number may be typed in or selected from a displayed directory at any of the stations located throughout the factory to instantaneously retrieve and display the job information at the user's location. The ability to instantaneously retrieve such information is aided by the use of communications network 26 and the storage of the design and information at a centrally located

database, such as database 30.

CLPR:
31. A system according to claim 30, wherein said face detecting system further comprises a system for generating a loop tree based on said outward loop defined by said initial linked list of entities and said inner loops defined by said additional linked list of entities.

CLPR:
32. A system according to claim 31, wherein said face detecting system detects said faces of said part based on said loop tree and the sequence of boundaries defined by said initial linked list entities and said additional linked list of entities.

CLPR:
92. A system according to claim 91, wherein said face detecting means further comprises means for generating a loop tree based on said outward loop defined by said initial linked list of entities and said inner loops defined by said additional linked list of entities.

CLPR:
93. A system according to claim 92, wherein said face detecting means detects said faces of said part based on said loop tree and the sequence of boundaries defined by said initial linked list entities and said additional linked list of entities.

5/11.0



## Document Number 20

Entry 20 of 20 File: USPT Jan 10, 1978

US-PAT-NO: 4068298
DOCUMENT-IDENTIFIER: US 4068298 A

TITLE: Information storage and retrieval system
DATE-ISSUED: January 10, 1978

INVENTOR-INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY |
|---|---|---|---|---|
| Dechant; Thomas Edward | Burton | OH | N/A | N/A |
| Glaser; Edward Lewis | Cleveland Heights | OH | N/A | N/A |
| Pitt; Paul Eldred | Malibu | CA | N/A | N/A |
| Way, III; Frederick | Cleveland Heights | OH | N/A | N/A |

ASSIGNEE INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY | TYPE CODE |
|---|---|---|---|---|---|
| Systems Development Corporation | Santa Monica | CA | N/A | N/A | 02 |

APPL-NO: 5/ 637511
DATE FILED: December 3, 1975

INT-CL: [2] G06F 7/00, G06F 15/40, G06F 9/06, H03K 13/00
US-CL-ISSUED: 364/200; 364/300, 340/347DD
US-CL-CURRENT: 707/3; 341/63, 341/76, 364/222.2, 364/222.81, 364/222.82, 364/225, 364/225.1, 364/225.4, 364/232.2, 364/232.7, 364/232.93, 364/238.4, 364/243, 364/243.1, 364/244, 364/245, 364/245.1, 364/246, 364/246.2, 364/248.5, 364/249, 364/249.1, 364/253, 364/253.1, 364/254.9, 364/256.8, 364/259, 364/259.2, 364/260.4, 364/260.6, 364/DIG.1
FIELD-OF-SEARCH: 340/347DD, 364/200, 364/300

REF-CITED:

U.S. PATENT DOCUMENTS

| PAT-NO | ISSUE-DATE | PATENTEE-NAME | US-CL |
|---|---|---|---|
| 3576534 | April 1971 | Steinberger | 340/146.3 |
| 3603937 | September 1971 | Loizides et al. | 340/172.5 |
| 3612660 | October 1971 | Miller | 340/347DD |
| 3618027 | November 1971 | Feng | 364/900 |
| 3643226 | February 1972 | Loizides et al. | 340/172.5 |
| 3651483 | March 1972 | Clark et al. | 340/172.5 |
| 3689915 | September 1972 | DeClerck et al. | 340/347DD |
| 3697950 | October 1972 | Low et al. | 340/146.1AQ |
| 3697951 | October 1972 | Bartholomew et al. | 340/146.3Q |
| 3793513 | February 1974 | Kaneko | 235/152 |
| 3821711 | June 1974 | Elam et al. | 340/347DD |
| 3938105 | February 1976 | Lechner | 340/172.5 |

OTHER PUBLICATIONS

James L. Massey, "Shift-Register Synthesis and BCH Decoding", IEEE Transactions on Information Theory, vol. IT-15, No. 1, Jan. 1969, pp. 122-126.
Lawrence T. Fisher, "Unateness Properties of AND-Exclusive-OR Logic Circuits", IEEE Transactions on Computers, vol. C-23, No. 2, Feb. 1974, pp. 166-172.
E. Henry Beitz, "A Set-Theoretic View of Data-Base Representation", ACM Sigmod Workshop on Data Description, Access and Control, ACM Sigfidet, Apr. 1974, pp. 478-494.
E. Henry Beitz, "Sets as A Model for Data Base Representation; Much Ado About Something", ACM Regional Conference, Pacific, 1975, pp. 80-84.
B. A. Marron and P. A. D. DeMaine, "Automatic Data Compression", Communications of the ACM, vol. 10, No. 11, Nov. 1967, pp. 711-715.
P. A. D. DeMaine, K. Kloss, B. A. Marron, "The Solid System. II. Numeric Compression, and the Solid System. III. Alphanumeric Compression", pp. 1-25, 27-42, NBS Technical Note 413, Superintendent of Documents, U.S. Government Printing Office, Washington, D. C. 20402, Aug. 15, 1967.
W. D. Hagamen, D. J. Linden, H. S. Long, J. C. Weber, "Encoding Verbal Information as Unique Numbers", IBM Syst. J, No. 4, 1972, pp. 278-315.

Jon Louis Bentley, "Multidimensional Binary Search Trees Used for Associative Searching", 1975 ACM Student Award Paper, Communications of the ACM, vol. 18, No. 9, Sept. 1975, pp. 509-517.
Bryant et al., "GIS and File Management", Proceedings of the 21st National Conference of the ACM, 1966, pp. 97-107, L71400874, 444/1.
H. Ling, F. P. Palermo, "Block-Oriented Information Compression", IBM J Res. Develop., Mar. 1975, pp. 141-145.
P. A. D. DeMaine and B. A. Marron, "The Solid System. I. A Method For Organizing and Searching Files, and the Solid System. II. Alphanumeric Compression", pp. 243-282, George Schecter (ed.) Information Retrieval-A Critical View, Thompson Book Company, Washington, D. C., 1967.

ART-UNIT: 237
PRIMARY-EXAMINER: Zache; Raulfe B.
ATTY-AGENT-FIRM: Christie, Parker & Hale

ABSTRACT:

Data processing information storage and retrieval system having a memory. A number of modules are interconnected with the memory. Encode and decode modules operate in conjunction with the memory for compacting and expanding data. A revolve module in association with a

delta module and a memory enable coded signals to be transferred into a number of unique but equivalent and related signals. A seed module enables the shortest of the equivalent signals to be located. A change module enables any one of the equivalent signals to be updated. An output module causes an equivalent signal to be converted back to the original signal representation. Pipe and brightness modules perform a discrimination function on stored information. The data processor includes programs which by unique means and methods structure and retrieve data from the data base. The retrieval may be based on an inexact match between events and entries of a request and the structured data base.

393 Claims, 160 Drawing figures

Document Number 20

Entry 20 of 20 File: USPT Jan 10, 1978

DOCUMENT-IDENTIFIER: US 4068298 A
TITLE: Information storage and retrieval system

BSPR:
Because of the differences in the techniques of the inquiry process in traditional and in holotropic information storage and retrieval systems, the structure of the latter may be vastly different. In traditional retrieval information storage and retrieval systems, an inquiry can be rejected because it contains an unallowable descriptor, or because something is misspelled, or because the parts are ordered improperly, or because the inquiry is not framed according to the specifications. Thus, an inquiry can be rejected regardless of whether the information it asked for is actually in the data base. In a holotropic data storage and retrieval system, no inquiry need ever be rejected for such reasons. The only sense in which an inquiry needs to be "rejected" at all by a holotropic information storage and retrieval system is that it fails to retrieve. In other words, the data base does not contain anything which matches the inquiry at the specified level of exactness. If this happens, the user is told whether or not a change in exactness will retrieve an item, and if so, the setting.

DEPR:
All incoming information to the DPM system is restructured by the MINI COMPUTER into a layered data base in its main memory. Each layer is a logical entity or a group of entities called "events". Each of these events is separated by a delimiter from a set of delimiters for the layer. The group of events between two subsequent delimiters is referred to as an "entry". Layering is hierarchical in that the higher level layers encompass the lower level layers. For example, if one were to structure contextual data base, the following levels may exist: layer 3 consisting of sentences; layer 2 consisting of phrases; layer 1 consisting of words; and layer 0 consisting of letters. Each layer has appropriate and distinct delimiters. However for purposes of illustration only a two layer system is specifically disclosed. One layer is for words and the second for sentences.

DEPR:
At the outset, it should be kept in mind that occurrence vectors represent a series of occurrence values out of a larger set of incrementally ordered possible occurrence values or event-times. Occurrence vectors are stored, retrieved and processed such that the highest numbered occurrence value is first. The highest numbered occurrence value identifies the most recent occurrence in the event-time domain. The lowest numbered entry, and hence the entry farthest back in event-time, is stored, retrieved and processed last. Examples of delimiter and event occurrence vectors (in absolute coded form) are shown at "" and "T" of Table 2. This form of information representation is quite important to an understanding of the ENCODE

MODULE embodiment about to be described and with respect to each of the other module embodiments about to be described.

DEPR:
FIG. 79C depicts the generalized operation of the disclosed data base which is shown in FIG. 121 and which is described in detail in Section XXXIII. FIG. 79C depicts the hierarchical arrangements of the programs and subroutines which comprise the data base system.

DEPR:
Consider now the current data base structures and the data base structure according to the present invention in light of the aforegoing definition. Regardless of their physical characteristics, presently known data bases can be thought of as systems which conceptually transform input data into a linear event-time domain. As each basic unit (e.g., character, letter, number, or other symbol), called an event, is entered into the system it is conceptually assigned a linear positional value which corresponds to the next state of some clock or sequentially ordering mechanism. This concept of the prior art data base system is depicted in FIG. 114. With a data base conceived in this fashion, the problem of retrieving information can be related to the problem of anwering the following question: Where in this linear sequence do the events of the request occur? If the ordering mechanism for the events in the data base is considered to be a clock, and the positional values are considered as clock times, the question becomes: At what clock time in the sequence do the events of the request occur? The question can be answered by linearly searching the data base until the response to the request is found. With large data bases this approach violates the preceding definition that information is to be retrieved with speed.

DEPR:
A further way in which the present invention speeds up a search is to store only new information. If information being entered into the data base is already contained in the data base there is no need to store it again. Therefore, according to the present invention, a hierarchical structuring of layers is employed which will permit redundancy to be squeezed out from the input data. In this way the information is compressed, less physical storage is used, and less search time is required to retrieve data. Such an arrangement is depicted by way of example in Tables 60B and 60C described hereinabove.

DEPR:
Also, the concept of the invention can be extended to include delimiters to squeeze out redundancy on layer 1. Thus, layers 0 and 1 identify letters and sentences. A layer 2 could be added to identify paragraphs or chapters, etc. The process is recursive and need only stop when there is no more redundancy that can be squeezed out from the system. The result of this process provides a hierarchical ordering of layers bound together by delimiters which attach the entries of a lower layer to events on the next higher layer. These delimiters then can be considered as the glue which holds the layers together. Delimiters may be implicit in that the Nth entry of a layer may implicitly point to the Nth event on the next higher layer. It should also be noted that no pointer storage is needed. Additionally, delimiters may be explicit in that there may be a table which relates each entry on a layer with events on other layers. The structure of this table can vary from a simple pointer table to a layer itself.

CLPV:
a. forming a request represented by parts, the parts including lower level entry parts which represent at least one higher level entry part, each lower level entry part having at least one event part which is represented by at least one coded event signal, in the

request the lower level entry parts and event parts being ordered in order of occurrence, the at least one higher level entry part corresponding to entries in the higher layer, lower level entry parts corresponding to entries in the lower layer (and events in the higher layer) and event parts corresponding to events in the lower layer;

CLPV:
a. means for forming a request represented by parts, the parts including lower level entry parts which represent at least one higher level entry part, each lower level entry part having at least one event part which is represented by at least one coded event signal, in the request the lower level entry parts and event parts being ordered in order of occurrence, the at least one higher level entry part corresponding to entries in the higher layer, lower level entry parts corresponding to entries in the lower layer (and events in the higher layer), and event parts corresponding to events in the lower layer;

ORPL:
Jon Louis Bentley, "Multidimensional Binary Search Trees Used for Associative Searching", 1975 ACM Student Award Paper, Communications of the ACM, vol. 18, No. 9, Sept. 1975, pp. 509-517.

5/11.00

# WEST

Help | Logout

Main Menu | Search Form | Posting Counts | Show S Numbers | Edit S Numbers

Generate Collection

**Search Results** - Record(s) 1 through 10 of 20 returned.

☐ 1. Document ID: US 5978516 A

Entry 1 of 20 File: USPT Nov 2, 1999

US-PAT-NO: 5978516
DOCUMENT-IDENTIFIER: US 5978516 A
TITLE: Method for checking convergence in fractal image coding

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWIC | Image

☐ 2. Document ID: US 5971589 A

Entry 2 of 20 File: USPT Oct 26, 1999

US-PAT-NO: 5971589
DOCUMENT-IDENTIFIER: US 5971589 A
TITLE: Apparatus and method for managing and distributing design and manufacturing information throughout a sheet metal production facility

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWIC | Image

☐ 3. Document ID: US 5940083 A

Entry 3 of 20 File: USPT Aug 17, 1999

US-PAT-NO: 5940083
DOCUMENT-IDENTIFIER: US 5940083 A
TITLE: Multi-curve rendering modification apparatus and method

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWIC | Image

☐ 4. Document ID: US 5890117 A

Entry 4 of 20 File: USPT Mar 30, 1999

US-PAT-NO: 5890117
DOCUMENT-IDENTIFIER: US 5890117 A
TITLE: Automated voice synthesis from text having a restricted known informational content

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWIC | Image

☐ 5. Document ID: US 5886897 A

Entry 5 of 20 File: USPT Mar 23, 1999

US-PAT-NO: 5886897
DOCUMENT-IDENTIFIER: US 5886897 A
TITLE: Apparatus and method for managing and distributing design and manufacturing information throughout a sheet metal production facility

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWIC | Image

☐ 6. Document ID: US 5867649 A

Entry 6 of 20 File: USPT Feb 2, 1999

US-PAT-NO: 5867649
DOCUMENT-IDENTIFIER: US 5867649 A
TITLE: Dance/multitude concurrent computation

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWIC | Image

☐ 7. Document ID: US 5864482 A

Entry 7 of 20 File: USPT Jan 26, 1999

US-PAT-NO: 5864482
DOCUMENT-IDENTIFIER: US 5864482 A
TITLE: Apparatus and method for managing distributing design and manufacturing information throughout a sheet metal production facility

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWIC | Image

☐ 8. Document ID: US 5857184 A

Entry 8 of 20 File: USPT Jan 5, 1999

US-PAT-NO: 5857184
DOCUMENT-IDENTIFIER: US 5857184 A
TITLE: Language and method for creating, organizing, and retrieving data from a database

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWIC | Image

☐ 9. Document ID: US 5842212 A

Entry 9 of 20 File: USPT Nov 24, 1998

US-PAT-NO: 5842212
DOCUMENT-IDENTIFIER: US 5842212 A
TITLE: Data modeling and computer access record memory

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWIC | Image

☐ 10. Document ID: US 5828575 A

Entry 10 of 20 File: USPT Oct 27, 1998

US-PAT-NO: 5828575
DOCUMENT-IDENTIFIER: US 5828575 A
TITLE: Apparatus and method for managing and distributing design and manufacturing information throughout a sheet metal production facility

**Search Results** - Record(s) 11 through 20 of 20 returned.

☐ 11. Document ID: US 5825936 A

Entry 11 of 20 File: USPT Oct 20, 1998

US-PAT-NO: 5825936
DOCUMENT-IDENTIFIER: US 5825936 A
TITLE: Image analyzing device using adaptive criteria

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWIC | Image

☐ 12. Document ID: US 5774525 A

Entry 12 of 20 File: USPT Jun 30, 1998

US-PAT-NO: 5774525
DOCUMENT-IDENTIFIER: US 5774525 A
TITLE: Method and apparatus utilizing dynamic questioning to provide secure access control

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWIC | Image

☐ 13. Document ID: US 5619630 A

Entry 13 of 20 File: USPT Apr 8, 1997

US-PAT-NO: 5619630
DOCUMENT-IDENTIFIER: US 5619630 A
TITLE: Apparatus for producing exploded view and animation of assembling and method thereof

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWIC | Image

☐ 14. Document ID: US 5602993 A

Entry 14 of 20 File: USPT Feb 11, 1997

US-PAT-NO: 5602993
DOCUMENT-IDENTIFIER: US 5602993 A
TITLE: Method and system for revising data in a distributed data communication system

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWIC | Image

☐ 15. Document ID: US 5583917 A

Entry 15 of 20 File: USPT Dec 10, 1996

US-PAT-NO: 5583917
DOCUMENT-IDENTIFIER: US 5583917 A
TITLE: Method and arrangement for semipermanent storage of a service profile in personal communication systems

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWIC | Image

☐ 16. Document ID: US 5311429 A

Entry 16 of 20 File: USPT May 10, 1994

US-PAT-NO: 5311429
DOCUMENT-IDENTIFIER: US 5311429 A
TITLE: Maintenance support method and apparatus for natural language processing system

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWIC | Image

☐ 17. Document ID: US 5220657 A

Entry 17 of 20 File: USPT Jun 15, 1993

US-PAT-NO: 5220657
DOCUMENT-IDENTIFIER: US 5220657 A
TITLE: Updating local copy of shared data in a collaborative system

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWIC | Image

☐ 18. Document ID: US 5008853 A

Entry 18 of 20 File: USPT Apr 16, 1991

US-PAT-NO: 5008853
DOCUMENT-IDENTIFIER: US 5008853 A
TITLE: Representation of collaborative multi-user activities relative to shared structured data objects in a networked workstation environment

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWIC | Image

☐ 19. Document ID: US 4591983 A

Entry 19 of 20 File: USPT May 27, 1986

US-PAT-NO: 4591983
DOCUMENT-IDENTIFIER: US 4591983 A
TITLE: Hierarchical knowledge system

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWIC | Image

☐ 20. Document ID: US 4068298 A

Entry 20 of 20 File: USPT Jan 10, 1978

US-PAT-NO: 4068298
DOCUMENT-IDENTIFIER: US 4068298 A
TITLE: Information storage and retrieval system

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWIC | Image

Generate Collection

| Terms | Documents |
|---|---|
| l1 and domain$1 | 20 |

Display 10 Documents including document number 20

**Display Format:** TI Change Format

Main Menu | Search Form | Posting Counts | Show S Numbers | Edit S Numbers

Help | Logout

**Search Results -**

| Terms | Documents |
|---|---|
| l1 and domain$1 | 20 |

**Database:** US Patents Full-Text Database

Refine Search: `l1 and domain$1`

**Search History**

| DB Name | Query | Hit Count | Set Name |
|---|---|---|---|
| USPT | l1 and domain$1 | 20 | L3 |
| USPT | 1 and domain$1 | 52776 | L2 |
| USPT | #S653 | 138 | L1 |

**Search Results** - Record(s) 1 through 10 of 17 returned.

☐ 1. Document ID: US 6055516 A

Entry 1 of 17 File: USPT Apr 25, 2000

US-PAT-NO: 6055516
DOCUMENT-IDENTIFIER: US 6055516 A
TITLE: Electronic sourcing system

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWIC | Image

☐ 2. Document ID: US 6023683 A

Entry 2 of 17 File: USPT Feb 8, 2000

US-PAT-NO: 6023683
DOCUMENT-IDENTIFIER: US 6023683 A
TITLE: Electronic sourcing system and method

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWIC | Image

☐ 3. Document ID: US 6009406 A

Entry 3 of 17 File: USPT Dec 28, 1999

US-PAT-NO: 6009406
DOCUMENT-IDENTIFIER: US 6009406 A
TITLE: Methodology and computer-based tools for re-engineering a custom-engineered product line

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWIC | Image

☐ 4. Document ID: US 6003012 A

Entry 4 of 17 File: USPT Dec 14, 1999

US-PAT-NO: 6003012
DOCUMENT-IDENTIFIER: US 6003012 A
TITLE: Methodology and computer-based tools for design, production and sales of customized switchboards

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWIC | Image

☐ 5. Document ID: US 5991728 A

Entry 5 of 17 File: USPT Nov 23, 1999

US-PAT-NO: 5991728
DOCUMENT-IDENTIFIER: US 5991728 A
TITLE: Method and system for the tracking and profiling of supply usage in a health care environment

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWIC | Image

☐ 6. Document ID: US 5878401 A

Entry 6 of 17 File: USPT Mar 2, 1999

US-PAT-NO: 5878401
DOCUMENT-IDENTIFIER: US 5878401 A
TITLE: Sales and inventory method and apparatus

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWIC | Image

☐ 7. Document ID: US 5877961 A

Entry 7 of 17 File: USPT Mar 2, 1999

US-PAT-NO: 5877961
DOCUMENT-IDENTIFIER: US 5877961 A
TITLE: Electronic support work station and method of operation

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWIC | Image

☐ 8. Document ID: US 5712989 A

Entry 8 of 17 File: USPT Jan 27, 1998

US-PAT-NO: 5712989
DOCUMENT-IDENTIFIER: US 5712989 A
TITLE: Just-in-time requisition and inventory management system

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWIC | Image

☐ 9. Document ID: US 5671362 A

Entry 9 of 17 File: USPT Sep 23, 1997

US-PAT-NO: 5671362
DOCUMENT-IDENTIFIER: US 5671362 A
TITLE: Materials monitoring systems, materials management systems and related methods

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWIC | Image

☐ 10. Document ID: US 5615109 A

Entry 10 of 17 File: USPT Mar 25, 1997

US-PAT-NO: 5615109
DOCUMENT-IDENTIFIER: US 5615109 A
TITLE: Method of and system for generating feasible, profit maximizing requisition sets

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWIC | Image

Generate Collection

**Search Results** - Record(s) 11 through 17 of 17 returned.

☐ 11. Document ID: US 5570291 A

Entry 11 of 17 File: USPT Oct 29, 1996

US-PAT-NO: 5570291
DOCUMENT-IDENTIFIER: US 5570291 A
TITLE: Custom product estimating and order processing system

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWIC | Image

☐ 12. Document ID: US 5515266 A

Entry 12 of 17 File: USPT May 7, 1996

US-PAT-NO: 5515266
DOCUMENT-IDENTIFIER: US 5515266 A
TITLE: Textile spinning machine management system

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWIC | Image

☐ 13. Document ID: US 5216612 A

Entry 13 of 17 File: USPT Jun 1, 1993

US-PAT-NO: 5216612
DOCUMENT-IDENTIFIER: US 5216612 A
TITLE: Intelligent computer integrated maintenance system and method

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWIC | Image

☐ 14. Document ID: US 5023438 A

Entry 14 of 17 File: USPT Jun 11, 1991

US-PAT-NO: 5023438
DOCUMENT-IDENTIFIER: US 5023438 A
TITLE: Portable data input apparatus with different display modes

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWIC | Image

☐ 15. Document ID: US 4803634 A

Entry 15 of 17 File: USPT Feb 7, 1989

US-PAT-NO: 4803634
DOCUMENT-IDENTIFIER: US 4803634 A
TITLE: Production process control system in newspaper printing

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWIC | Image

☐ 16. Document ID: US 4254329 A

Entry 16 of 17 File: USPT Mar 3, 1981

US-PAT-NO: 4254329
DOCUMENT-IDENTIFIER: US 4254329 A
TITLE: Microfiche information retrieval and control system utilizing machine readable microfiche and visually readable microfiche

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWIC | Image

☐ 17. Document ID: US 4181954 A

Entry 17 of 17 File: USPT Jan 1, 1980

US-PAT-NO: 4181954
DOCUMENT-IDENTIFIER: US 4181954 A
TITLE: Computer-aided graphics system including a computerized material control system and method of using same

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWIC | Image

Generate Collection

| Terms | Documents |
|---|---|
| l18 and l19 | 17 |

Display 10 Documents including document number 17

**Display Format:** TI Change Format

Main Menu | Search Form | Posting Counts | Show S Numbers | Edit S Numbers

Help | Logout

logic from existing **stock** control, **ordering** , **parts** management, sales order, order **checking** and other systems and putting it up on an extranet, linking a company with its **suppliers** , distributors and customers. Written in C, it uses HTML templates, ActiveX, runs mostly on Windows...

...it is now seeking a strategic partner and funding to help it expand beyond the **manufacturing** sector it exclusively services at present into healthcare and mobile markets. It will also begin...

**6/8,K/3 (Item 2 from file: 275)**
DIALOG(R)File 275:(c) 2000 The Gale Group. All rts. reserv.

01212179 SUPPLIER NUMBER: 04770683 (USE FORMAT 7 OR 9 FOR FULL TEXT)
**Porsche runs on smooth network. (connectivity section)**
April 14, 1987
WORD COUNT: 1703 LINE COUNT: 00139

SPECIAL FEATURES: illustration; photograph; table; chart
COMPANY NAMES: Porsche Cars North America Inc.--Communication systems
DESCRIPTORS: Networks; Minicomputer; Automobile Industry; Distribution Management; Software; Personal Computers; Network Architecture
SIC CODES: 3711 Motor vehicles and car bodies
TRADE NAMES: IBM System/38 (Minicomputer)--Usage
FILE SEGMENT: CD File 275

... values (except for parts ordering; there are too many possible parts to maintain a local **inventory** ). Thus, "You can't order a sunroof on a convertible," said Mr. Verheul. The data...

...disk. Records have a fixed length of 248 bytes, a structure inherited from the GEISCO **store** -and-collect communications system.
One example of the system's keep-it-simple philosophy is...

**6/8,K/4 (Item 1 from file: 636)**
DIALOG(R)File 636:(c) 2000 The Gale Group. All rts. reserv.

02721867 Supplier Number: 45521180 (USE FORMAT 7 FOR FULLTEXT)
**PARTCO LAUNCHES AUTOPARTNER - ITS NEW COMPUTER GARAGE MANAGEMENT SYSTEM - AT ATS '95**
May 4, 1995
Word Count: 1178
PUBLISHER NAME: M2 Communications
INDUSTRY NAMES: BUSN (Any type of business); INTL (Business, International)

produced detailing parts to be fitted, areas to be **checked** , and relevant job times. Credit notes and invoices - again linked to specific quotations if required...

...time. A garage could be printing invoices at the same time as booking jobs and **ordering** **parts** - with another screen viewing management information. Partco is marketing Autopartner with an extensive programme of ...

...video, garages should telephone 0181 744 8444. Note to editors: Partco is the largest independent **supplier** of parts and equipment to the UK garage trade , with over 240 branches nationwide. The company employs over 2,000 people and has a fleet of around 670 vehicles. The **Retail** Motor Industry Federation is the automotive industry's largest trade association with some 12,000...

...44 1327 342020 M2 COMMUNICATIONS DISCLAIMS ALL LIABILITY FOR INFORMATION PROVIDED WITHIN M2 PRESSWIRE. DATA **SUPPLIED** BY NAMED PARTY/PARTIES.

**6/8,K/5 (Item 1 from file: 16)**
DIALOG(R)File 16:(c) 2000 The Gale Group. All rts. reserv.

01201987 Supplier Number: 41381372 (USE FORMAT 7 FOR FULLTEXT)
**Kroger combos power Rx, non-food drive**
June 11, 1990
Word Count: 529
PUBLISHER NAME: Lebhar-Friedman, Inc.
COMPANY NAMES: *Kroger
EVENT NAMES: *220 (Strategy & planning)
GEOGRAPHIC NAMES: *1USA (United States)
PRODUCT NAMES: *5411100 (Supermarkets)
INDUSTRY NAMES: BUSN (Any type of business); DRUG (Pharmaceuticals and Cosmetics); RETL (Retailing)
NAICS CODES: 44511 (Supermarkets and Other Grocery (except Convenience) Stores)
SPECIAL FEATURES: COMPANY
ADVERTISING CODES: 55 Company Planning/Goals; 81 Demographics

... a host of consumer segments.
"Kroger's opportunity in the '90s is to orchestrate our **stores** , our systems and talents to reach customers as individuals," said Pichler. "Kroger is implementing computer systems that expedite the **ordering** of **products** ; keep our shelves fully **stocked** ; identify the product preferences of individual customers; offer values designed to meet the particular preferences of loyal customers; and improve the speed of **checkout** ."
Among new ideas being tested, Pichler said, are instore information systems for shoppers; shopping carts...

**6/8,K/6 (Item 1 from file: 160)**
DIALOG(R)File 160:(c) 1999 The Gale Group. All rts. reserv.

00978885
**Computer-guided ordering (CGO) help retailers avoid errors.**
December, 1983

PRODUCT: *Retail Stores (5200110)
EVENT: *General Services (26)
COUNTRY: *United States (1USA)

... held data entry unit. Following completion of the order, it is put into an in- **store** minicomputer, where it will be edited, subjected to reviews and analyses, and then transmitted to...

... warehouse. At this point, each order is logged in a file for future reference. By **checking** an order history file, managers can ask the in- **store** computer for a list of those products in the department that are authorized for sale but have not been ordered recently. Hence, the manager can **check** for products that have been lost from the shelf using the list. A computer-guided **ordering** system can report sale program **items** that have not been ordered. This report can catch potential mistakes before an order is...
... can enforce product restrictions, products each department manager believed should not be carried in the **store** in the first place. The CGO can be used to force out backroom unwanted **inventory** and reduce the labor that is associated with ordering. CGO is the first **retail** automation step in an application development process that will end with a computer generated order...

**6/8,K/7 (Item 2 from file: 160)**
DIALOG(R)File 160:(c) 1999 The Gale Group. All rts. reserv.

00790066
**Boeing and Procter & Gamble face possible antitrust charges by the Federal Trade Commission.**

April 26, 1982

COMPANY:
*Boeing DUNS: 00-925-6819 TICKER: BA (NYSE) CUSIP: 097023
Procter & Gamble DUNS: 00-131-6827 TICKER: PG (NYSE) CUSIP: 742718

PRODUCT: *Aircraft & Parts (3720000); Soaps & Detergents (2841000)
EVENT: *Government Regulation (cont) (94)
COUNTRY: *United States (1USA)

... The Boeing investigation focuses on claims that thhe aerospace corporation is curtailing competition in the **manufacture** and sale of aircraft replacement parts. The FTC's Seattle office is **checking** complaints that Boeing has created some unspecific anticompetitive conditions and restraints on its clients and **suppliers** . In the Procter & Gamble examination, the Boston office is **checking** claims that the company has conspired with several of its long time customers, conventional retailers, against big discount drug **stores** and other discount outlets. That probe focuses on complaints that P&G will not sell bargain-priced items to wholesalers and retailers that won't continue **ordering** those **products** when they are not **suppled** at a discount.

**6/8,K/8 (Item 1 from file: 148)**
DIALOG(R)File 148:(c)2000 The Gale Group. All rts. reserv.

09645590 SUPPLIER NUMBER: 17518075 (USE FORMAT 7 OR 9 FOR FULL TEXT)
**More catalogs go online.(electronic purchasing catalogs)**
Sep 21, 1995
WORD COUNT: 1018 LINE COUNT: 00092

SPECIAL FEATURES: photograph; table; illustration
INDUSTRY CODES/NAMES: BUSN Any type of business; TRAN Transportation, Distribution and Purchasing
DESCRIPTORS: Catalogs--Usage; Purchasing departments--Automation
PRODUCT/INDUSTRY NAMES: 7375000 (Database Providers)
SIC CODES: 7375 Information retrieval services; 7372 Prepackaged software
FILE SEGMENT: TI File 148

... prices, and catalog. It is capable of setting dollar limits on orders or to restrict **ordering** of specific **items** , and includes a report writer with sophisticated reports on purchasing, usage, and budgets. For every item, CEO displays negotiated price, list price, **manufacturer** , and catalog page number. It flags high-use items or those that are new, recycled, or provided by minority **suppliers** . It accepts procurement cards for payment. CEO is EDI-ready. It connects to Boise Cascade **SupplyLine** for on-line access to **check inventory** or account information. Three levels of passwords determines who can create or approve orders.
* Vogel...

**6/8,K/9 (Item 2 from file: 148)**
DIALOG(R)File 148:(c)2000 The Gale Group. All rts. reserv.

07229269 SUPPLIER NUMBER: 15311801 (USE FORMAT 7 OR 9 FOR FULL TEXT)
**Leybold-Inficon tackles product development. (Leybold-Inficon Inc.) (Management)**
March, 1994
WORD COUNT: 530 LINE COUNT: 00040

SPECIAL FEATURES: illustration; table
COMPANY NAMES: Leybold Inficon Inc.--Quality control
INDUSTRY CODES/NAMES: ELEC Electronics
DESCRIPTORS: Pumping machinery industry--Quality control; Production standards--Usage; Product development--Technique
PRODUCT/INDUSTRY NAMES: 3569293 (Industrial Vacuums)
SIC CODES: 3561 Pumps and pumping equipment; 3599 Industrial machinery,

not elsewhere classified; 3589 Service industry machinery, not elsewhere classified
FILE SEGMENT: TI File 148

... for certification, we found that the ISO standard specified a number of requirements for design **verification** . Within the company, every group was doing some piece the right way, but no one...
...different departments looked at standardization favorably, recognizing that the existing structure led to problems in **manufacturing** . For example, **ordering** too many types of **parts** ruled out quantity discounts from **suppliers** , and sometimes it was difficult to figure out whether we should do assembly work in...

**6/8,K/10 (Item 3 from file: 148)**


04600528 SUPPLIER NUMBER: 08546130 (USE FORMAT 7 OR 9 FOR FULL TEXT)
**Kroger combos power Rx, non-food drive. (Kroger Co. combination food-drug stores; includes related article)**
June 11, 1990
WORD COUNT: 712 LINE COUNT: 00058

SPECIAL FEATURES: illustration; photograph
COMPANY NAMES: Kroger Co.--Management
INDUSTRY CODES/NAMES: DRUG Pharmaceuticals and Cosmetics; RETL Retailing
DESCRIPTORS: Drugstores--Management; Supermarkets--Management
SIC CODES: 5912 Drug stores and proprietary stores; 5411 Grocery stores
FILE SEGMENT: TI File 148

... a host of consumer segments.
"Kroger's opportunity in the '90s is to orchestrate our **stores** , our systems and talents to reach customers as individuals," said Pichler. "Kroger is implementing computer systems that expedite the **ordering** of **products** ; keep our shelves fully **stocked** ; identify the product preferences of individual customers; offer values designed to meet the particular preferences of loyal customers; and improve the speed of **checkout** ."
Among new ideas being tested, Pichler said, are instore information systems for shoppers; shopping carts...

**6/8,K/11 (Item 4 from file: 148)**


03913747 SUPPLIER NUMBER: 07594473 (USE FORMAT 7 OR 9 FOR FULL TEXT)
**To market, to market: job opportunities in marketing.**
Spring, 1989
WORD COUNT: 9371 LINE COUNT: 00774

SPECIAL FEATURES: illustration; table; photograph
INDUSTRY CODES/NAMES: BUS Business, General
DESCRIPTORS: Marketing--Vocational guidance; Employment--Forecasts
FILE SEGMENT: MI File 47

... assistance, and resolve problems.
Sales representatives may also perform many services for retailers, such as **checking** the **store** 's **stock** and **ordering** **items** that will be needed before the next visit. Some sales representatives help retailers improve their ordering and **inventory** systems and advise them about advertising, pricing, and displays. Sales workers who handle industrial machinery...

**6/8,K/12 (Item 5 from file: 148)**

03283250 SUPPLIER NUMBER: 05057448 (USE FORMAT 7 OR 9 FOR FULL TEXT)
**Man the pumps. (Adaptec Inc.) (company profile)**
July 13, 1987
WORD COUNT: 1115 LINE COUNT: 00089

SPECIAL FEATURES: illustration; portrait
COMPANY NAMES: Adaptec Inc.--Management
INDUSTRY CODES/NAMES: BUS Business, General
DESCRIPTORS: Computer peripherals industry--Management
NAMED PERSONS: Boucher, Lawrence--Management
SIC CODES: 3577 Computer peripheral equipment, not elsewhere classified
FILE SEGMENT: MI File 47

... to manage the results. There were no regular meetings among Adaptec's marketing, sales and **manufacturing** staffs to refine production forecasts. Finances were so disorganized that the company was not **verifying** shipments coming in against purchase orders. **Suppliers** were shipping parts Adaptec didn't need and back-**ordering** **parts** it did, and the company didn't know any better. Receivables were bumping 100 days...

**6/8,K/13 (Item 1 from file: 20)**


03546723
**PR Newswire California Summary, Tuesday, November 24 up to 10:00 AM PT**
November 24, 1998
WORD COUNT: 1265

... CA-Solectron-Europe (MILPITAS) Solectron Expands Its Low-Cost, High-Volume PCBA and Systems Assembly **Manufacturing** Capacity in Europe NYFNSI08 11/24/1998 04:07 r l bc-Holiday-HerbMarket (SACRAMENTO...

... f bc-CA-Redwood-Empire (SANTA ROSA) Redwood Empire Bancorp Declares Quarterly Dividends on Common **Stock** SFTU013 11/24/1998 06:30 r f bc-OR-Metro-One-Fast-500 (PORTLAND...

... r f bc-CA-Data-Dime-services (BELLEVUE) Data Dimensions to Provide Test Support and **Validation** Services to Countrywide Home Loans HSTU003 11/24/1998 07:59 r f bc-FL...3 GB Per Disk SFTU019 11/24/1998 08:30 r f bc-CA-Flextronics- **stock** (SAN JOSE) Flextronics International Announces Two-for-One **Stock** Split LATU033 11/24/1998 08:39 r s bc-CA-Notre-Dame-USC (LOS...

... DEL MAR) Shopping.com Announces the Grand Opening of the Internet's First Full Service **Retail** Destination Hub Site LATU006 11/24/1998 09:00 r f bc-CA-Esquire-Comm- **Stock** (SAN DIEGO) Esquire Communications Announces 1-For-2 Reverse **Stock** Split SFTU022 11/24/1998 09:00 r f bc-CA-@Home-shop-online (REDWOOD...

... 09:01 r f bc-CA-Lancer-Orthodontic (SAN MARCOS) Lancer Orthodontics Introduces New Internet **Ordering** System for its **Products** DATU003 11/24/1998 09:15 r f bc-CA-Transnational-agr (SAN FRANCISCO) Transnational ...

... Plus Web Creates and Publishes 20,000 Vendor Pages with 33,000 Products on AOL **Store** in Less Than 12 Hours NYTU034 11/24/1998 10:00 r f bc-NJ...

**6/8,K/14 (Item 2 from file: 20)**


02970721
**Three Companies With Three Needs Select One Solution**
September 30, 1998
WORD COUNT: 720

COMPANY NAMES: Dendrite International Inc

DESCRIPTORS: Equities-Market
COUNTRY NAMES/CODES: United States of America (US) ; United Kingdom (GB)
REGIONS: Americas; North America; Pacific Rim; Europe; European Union; Western Europe
PROVINCE/STATE: New Jersey
SIC CODES/DESCRIPTIONS: 7372 (Prepackaged Software)

...N.J.--(BUSINESS WIRE)--Sept. 29, 1998--Dendrite International, Inc. (NASDAQ/NMS:DRTE), the leading **supplier** of enterprise-wide sales force effectiveness solutions for the Pharmaceutical and Consumer Packaged Goods (CPG...

...Australia (SBPA) will be equipped with the latest in customer management technology. Reckitt & Colman Australia, **manufacturers** of over-the-counter pharmaceuticals, prescription medicine and personal care products, chose Dendrite for the...

... Electronic Territory Management system to assist in their ability to detail critical information and streamline **ordering** of SBPA **products** . According to John Cosgrove, National Sales Manager with SBPA, the new system has a number...

... to customer service, by providing the efficient method of compiling and processing pharmacy orders and **inventory** management." ABOUT DeNDRITE Over 40,000 sales representatives and their managers in over 150 major...

... Company's inception. Dendrite has two major divisions: the Healthcare Division, which is the largest **supplier** of sales force effectiveness solutions to the global pharmaceutical industry; and the Consumer Business Division...

... of Dendrite International, Inc. This press release may contain statements that, if they are not **verifiable** historical fact, may be viewed as forward-looking statements that could predict future events or...

**6/8,K/15 (Item 3 from file: 20)**


02901991
**Scientific-Atlanta and Wink Sign Agreement for Wink's Enhanced Broadcasting System to Operate on Scientific-Atlanta's Digital Cable Set-Tops**
September 23, 1998
WORD COUNT: 768

COUNTRY NAMES/CODES: United States of America)
REGIONS: North America
PROVINCE/STATE: Georgia
SIC CODES/DESCRIPTIONS: 3660 ( Communications Equipment); 4810 ( Telephone Communications)

... enhancements to their existing video programming and commercials. Viewers can respond by requesting information or **ordering products** through a remote control and a Wink-enabled set-top. Wink's Response Network aggregates...

... applications in the marketplace. Developers receive technical support services, software discounts and certification services to **verify** reliable operation on the cable network. The program is part of Scientific-Atlanta's strategy...
... operators, such as Century Communications, Charter Communications and InterMedia, and television and set-top terminal **manufacturers** . Wink has also formed strategic relationships with a number of broadcast and cable networks, such...

... representing 12 million cable subscribers. Scientific-Atlanta, Inc. (http://www.sciatl.com) is a leading **supplier** of broadband communications systems, satellite-based video, voice and data communications networks and

worldwide customer...

**6/8,K/16 (Item 1 from file: 47)**
DIALOG(R)File 47:(c) 2000 The Gale group. All rts. reserv.

02949145 SUPPLIER NUMBER: 04755183 (USE FORMAT 7 OR 9 FOR FULL TEXT)
**Porsche runs on smooth network. (North American division's communications system)**
April 14, 1987
WORD COUNT: 1703 LINE COUNT: 00139

SPECIAL FEATURES: illustration; photograph; table; chart
COMPANY NAMES: Porsche Cars North America Inc.--Communication systems
DESCRIPTORS: Automobile industry--Communication systems; Automobile dealers--Communication systems; Electronic mail systems--Usage
SIC CODES: 3711 Motor vehicles and car bodies; 5511 New and used car dealers
FILE SEGMENT: MI File 47

... by the dealership, and delivery querying.
The data entered for any of these applications is **checked** against a table of allowable values (except for parts **ordering** ; there are too many possible **parts** to maintain a local **inventory** ). Thus, "You can't order a sunroof on a convertible," said Mr. Verheul. The data...

...disk. Records have a fixed length of 248 bytes, a structure inherited from the GEISCO **store** -and-collect communications system.
One example of the system's keep-it-simple philosophy is...

**6/8,K/17 (Item 1 from file: 570)**
DIALOG(R)File 570:(c) 2000 The Gale Group. All rts. reserv.

01832324 Supplier Number: 57010317
**Completed information system puts 7-Eleven at forefront of convenience store technology.**
Sept, 1999
PUBLISHER NAME: NRF Enterprises Inc.
COMPANY NAMES: *Seven-Eleven Food Stores
PRODUCT NAMES: *5411300 (Convenience Stores)
INDUSTRY NAMES: BUSN (Any type of business); RETL (Retailing)
NAICS CODES: 44512 (Convenience Stores)
SPECIAL FEATURES: COMPANY

ABSTRACT:
Convenience **store** chain 7-Eleven has installed an **inventory** management and sales data system technology as part of its five-year program to aid its core business functions. The core business functions refer to accounting, cash control, cash payables, **inventory** control, **ordering** and **products** . The system, called the **Retail** Information System, will speed up **checkout** time for customers and capture point-of-sale information. This, in turn, will be used to better manage the **store** 's **inventory** , promote sales and improve accounting and merchandising procedures.
...
?show files;ds
File 15:ABI/INFORM(R) 1971-2000/May 17
(c) 2000 Bell & Howell
File 9:Business & Industry(R) Jul/1994-2000/May 17
(c) 2000 Resp. DB Svcs.
File 623:Business Week 1985-2000/May W1
(c) 2000 The McGraw-Hill Companies Inc
File 810:Business Wire 1986-1999/Feb 28
(c) 1999 Business Wire
File 275:Gale Group Computer DB(TM) 1983-2000/May 17
(c) 2000 The Gale Group
File 624:McGraw-Hill Publications 1985-2000/May 17